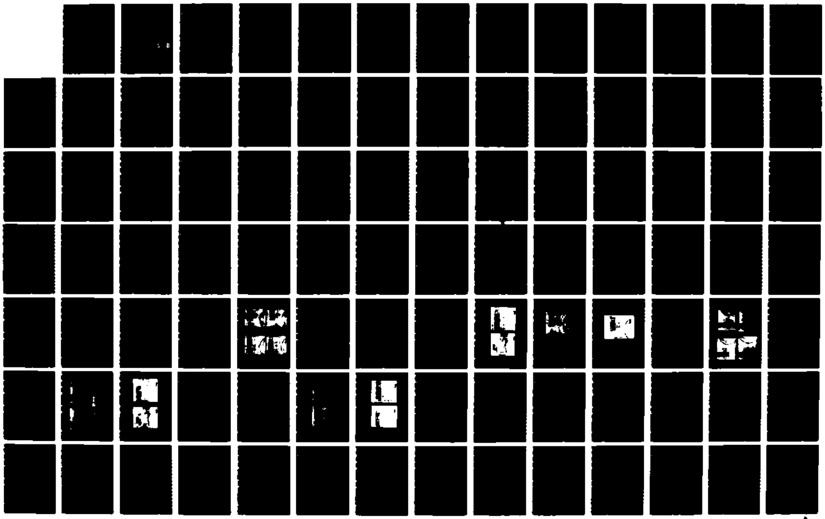
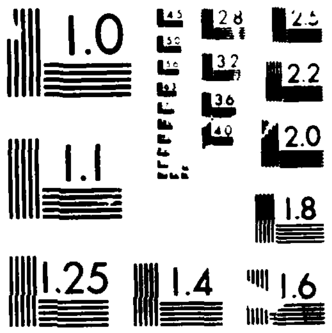
AD-A187 987 FIXED AND ADAPTIVE INTERFRAME IMAGE PREDICTION FOR REMOTE PILOTING APPLICATION(U) ILLINOIS UNIV AT URBANA COORDINATED SCIENCE LAB M G RAGHAVENDER OCT 87 1/1

UNCLASSIFIED UILU-END-87-2263 N00014-84-C-0149 F/G 25/5 NL



END



MICROCOPY RESOLUTION TEST CHART

1963 A

October 1987 DTIC FILE COPY UILU-ENG-87-2263
DC-98

(2)

COORDINATED SCIENCE LABORATORY
College of Engineering
Decision and Control Laboratory

FIXED AND ADAPTIVE INTERFRAME IMAGE PREDICTION FOR REMOTE PILOTING APPLICATION

Manorama Gollakota Raghavender

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS None
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) UILU-ENG-87-2263 (DC-98)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Coordinated Science Lab University of Illinois — 6b. OFFICE SYMBOL (If applicable) N/A	7a. NAME OF MONITORING ORGANIZATION Office of Naval Research
6c. ADDRESS (City, State, and ZIP Code) 1101 W. Springfield Ave. Urbana, IL 61801	7b. ADDRESS (City, State, and ZIP Code) 800 N. Quincy St. Arlington, VA 22217
8a. NAME OF FUNDING/SPONSORING ORGANIZATION Joint Services Electronics Program — 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER N00014-84-C-0149
8c. ADDRESS (City, State, and ZIP Code) 800 N. Quincy St. Arlington, VA 22217	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. | PROJECT NO. | TASK NO. | WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification) FIXED AND ADAPTIVE INTERFRAME IMAGE PREDICTION FOR REMOTE PILOTING APPLICATION

12. PERSONAL AUTHOR(S) MANORAMA GOLLAKOTA RAGHAVENDER

13a. TYPE OF REPORT Technical	13b. TIME COVERED FROM ______ TO ______	14. DATE OF REPORT (Year, Month, Day) Oct ~~Sept.~~ 1987	15. PAGE COUNT 93

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	image processing, fixed predictor, adaptive predictor, motion estimation, remote piloting,

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

Digital image processing has gained considerable importance due to its numerous applications in the aerospace, biomedical, commercial television and video teleconferencing systems. The availability of super-fast chips for digital data processing has made the hardware implementation of the image processing algorithms feasible for satellite applications due to the reduction achieved in weight, size and power consumption. A considerable amount of work done in the area of image processing has focused on coding, bandwidth compression and pattern recognition.

In this work, we focus on the estimation of image data using past statistics. We are interested in an on-line prediction of the next few frames of a video sequence using the available frames. The problem is that of parameter identification of a time-varying system using a priori knowledge. We apply estimation theory concepts and derive a fixed predictor as well as one that is adaptive, i.e., one which predicts frames by analyzing that data which was received most recently.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT [✓ UNCLASSIFIED/UNLIMITED] ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION Unclassified	
22a. NAME OF RESPONSIBLE INDIVIDUAL	22b. TELEPHONE (Include Area Code)	22c. OFFICE SYMBOL

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

FIXED AND ADAPTIVE INTERFRAME IMAGE PREDICTION FOR REMOTE PILOTING APPLICATION

BY

MANORAMA GOLLAKOTA RAGHAVENDER

B.Tech., G. B. Pant University of Agriculture and Technology, 1978

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 1987

Urbana, Illinois

ACKNOWLEDGMENTS

The author would like to express her sincere gratitude to her advisor, Professor W. R. Perkins, for his encouragement and guidance throughout the development of this thesis. Sincere thanks are also extended to Professor H. V. Poor for his interest in this work and the discussions that helped shape the final report. I would also like to thank Ms. Maria Woods of TRW, whose suggestions helped a lot in the simulations. Finally, deepest thanks are extended to Ms. Rose Harris for help in completing the various formalities.

TABLE OF CONTENTS

CHAPTER 1
INTRODUCTION

Digital image processing has gained considerable importance due to its numerous applications in the aerospace, biomedical, commercial television and video teleconferencing systems. The availability of super-fast chips for digital data processing has made the hardware implementation of the image processing algorithms feasible for satellite applications due to the reduction achieved in weight, size and power consumption. A considerable amount of work done in the area of image processing has focused on coding, bandwidth compression and pattern recognition.

In the area of image processing on board a satellite, the usual objectives are image enhancement, efficient encoding to reduce the transmission or storage capacity requirements and pattern recognition for the purpose of extraction of certain feature points. In this work, we focus on a different aspect of digital data processing. Here, we are concerned with the estimation of image data using past statistics. Specifically, we are interested in an on-line prediction of the next few frames of a video sequence using the available frames. The problem then is that of parameter identification of a time-varying system using a priori knowledge. For this purpose, we apply estimation theory concepts and derive a fixed predictor as well as one that is adaptive, i.e., one which predicts frames by analyzing that data which was received most recently. In the former, a fixed prediction algorithm is used whereas in the latter it is based on the most recent data. The specific application considered is that of a remotely piloted vehicle where a man-in-the-loop uses images relayed by a spacecraft in orbit for remotely maneuvering the vehicle. The prediction of the image data is expected to enhance the pilot's ability to maneuver the vehicle by compensating for the data which are either corrupted by channel noise or lost because of a temporary loss in the communication link. Otherwise, the estimates of the next frames impart added knowledge about the scene and the target movement, and the resulting smoothing effect is expected to aid significantly in the piloting operation.

A brief description of the other areas of research is presented below for a contrast with our objective. The area of image enhancement is concerned with restoring the quality of the pictures, which may be degraded because of a noisy satellite channel, or enhancing the contrast for a better scene interpretation. The results of image enhancement work done at the Jet Propulsion Laboratory for improving the picture quality of the pictures of the Moon and Mars are well known. Pattern recognition is another important area of research, and refers to extraction of patterns or other information from images. Its applications are in the area of biomedical engineering, automatic mapping of earth resources from satellite photographs, etc. Video bandwidth compression is an area that has received a lot of attention and is concerned with the problem of bandwidth constraint either for storage or transmission. For instance, the bandwidth available from a spacecraft for real-time transmission is severely limited due to the total weight, equipment size and power constraints. This necessitates compressing of the image data into a much smaller bandwidth, simultaneously minimizing any degradation in the received picture quality. Bandwidth compression techniques seek to achieve this by removing the redundancy inherent in an image, or a sequence of images, both in the space domain as well as the time domain so that the image can be represented by a smaller bandwidth. A considerable amount of work in the image processing area has focused on this problem. Seyler [1] describes a coding technique to reduce the channel capacity requirement. The applications of predictive coders and transform coders are well known. The former exploit spatial and temporal redundancies in the data, and the latter transform the image data into the frequency domain, and achieve compression by exploiting the fact that the human eye is sensitive only to changes in the lower frequency coefficients. Hybrid techniques [2] have also been widely applied since they use a combination of algorithms to achieve the best compromise between implementation complexity and performance. Adaptive compression techniques are important due to their ability to monitor their performance and inject a feedback term in their algorithm, to adapt themselves to changes in the scene statistics. This makes them more robust than their nonadaptive counterparts. Ericsson [3] reports good results when applying adaptive predictors rather than fixed predictors for bandwidth reduction via interframe coding. A survey of adaptive image coding techniques is given in [4]. For applications involving feedback control systems and bandwidth compression systems, great improvements in performance have been reported with the use of adaptive techniques. This was the motivation to apply this approach for image data prediction.

As stated above, this work addresses a different problem. Here, the dynamic estimation problem deals with the determination of image pixel intensities of a video frame based upon those of the frames already received, i.e., estimation of unknown, time-varying parameters using measurable data. The motivation for this work is as follows: A teleoperator-based remotely piloted spacecraft transmits video images obtained from the on-board cameras to a ground control station for scene and target interpretations. A man-in-the loop (the pilot on the ground control station) relies on the images transmitted from the spacecraft for maneuvering it near a target spacecraft for surveillance. In the case of a disabled spacecraft, the aims are rendezvousing and docking for the purpose of retrieval. In this application, knowledge of the next few frames of the video sequence could greatly enhance the pilot's perception of the scene and target motion, and thus aid significantly in the remote piloting operation.

An added motivation is due to the problem created by a temporary loss in the communication link between the spacecraft and the ground control station. As shown in Figure 1, the spacecraft in orbit modulates the digitized video signal onto a radio frequency (RF) carrier and transmits it to a communication satellite such as NASA's Telecommunications and Data Relay Satellite (TDRS). The TDRS receives the signal, amplifies it, remodulates it onto another RF carrier and transmits it to the ground control station. In a situation where the target spacecraft is spinning, the parent spacecraft will have to spin up to the same rate to be able to dock with the target. A spinning spacecraft would have to alternate between its antennas for transmission of the data as the relay satellite moves out of the field of view (FOV) of one antenna and into that of the other. The FOV of the satellite antennas is usually limited by the antenna size and weight constraints. It is conceivable that during a part of the rotation, the communication satellite may not be within the

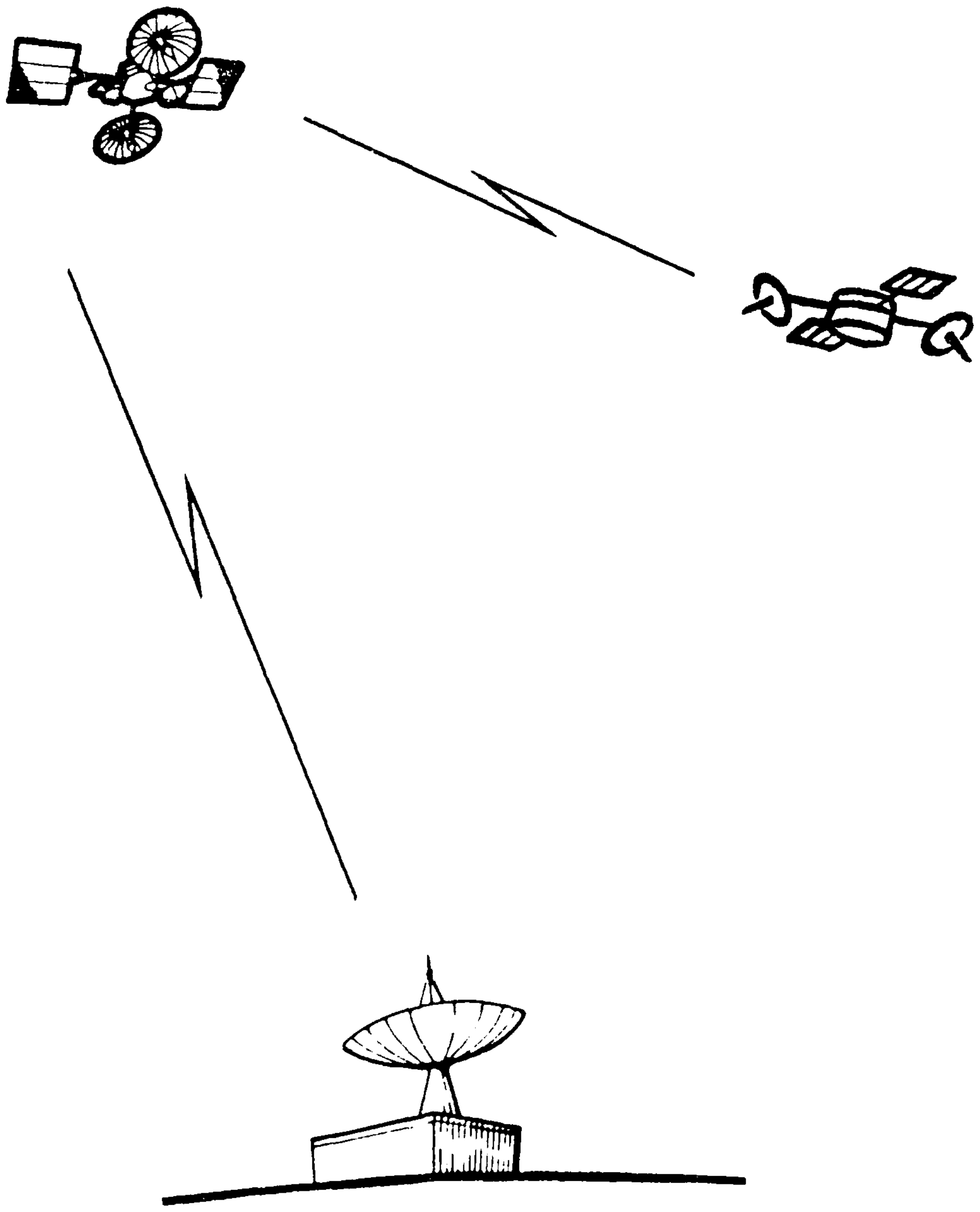


Figure 1. Spacecraft-to-earth station communication via a relay satellite.

FOV of either antenna. This would result in a temporary loss in the communication link resulting in the loss of a few frames. A poor link performance (bit error rate of the channel) could also potentially result in some loss of data, impairing the pilot's ability to maneuver the vehicle, resulting in either a loss of the mission, the spacecraft, or both. In such cases, reconstruction of the missing frames using the past statistics could avert a catastrophic failure. The problem then is not only filtering of the data based on the frames received, but also that of predicting the next few frames. Even if there were no missing frames, image data prediction offers smoothing of the data which would considerably increase the pilot's perception of the scene.

For the sake of image data restoration, the simplest solution may appear to be a frame refresh based upon the last frame received. This work seeks to exploit the statistical correlation between the pixels of adjacent frames of a video sequence for a more accurate prediction of the images. Specifically, a fixed and an adaptive predictor are utilized. Underlying both of these approaches is the problem of parameter estimation of time-varying parameters. For solving the prediction problem, we first represent the image sequence as a discrete-time linear state vector model. The challenge presented by this approach of using fixed and adaptive frame predictions of a video sequence based upon the past frames received is in modeling the scene dynamics and representing the image processing problem as a state vector model. When the system model is completely specified, standard parameter estimation techniques can be used for designing optimal predictors. In our case, however, the system model is not completely specified. The problem is compounded by the fact that the video image sequence is characterized by time-varying parameters rather than stationary ones. We approach this problem in the fixed predictor case by exploiting the inherent correlation of the adjacent pixels of a frame, and that of adjacent frames to derive the state vector model by assuming a fixed interframe and intraframe correlation. In the case of an adaptive predictor, we derive the interframe correlation from the set of frames received and assume that in the case of slow dynamics, which is typically the case in spacecraft-to-spacecraft docking situation, the same interframe correlation can be applied to the next set of frames. We intend to investigate if it is possible to obtain better results using these approaches than with simple frame refresh.

Chapter 2 gives the formulation of the image processing problem.

A truly adaptive predictor would have to take into account and compensate for the relative displacement between successive frames of a sequence. This is because the scene is actually nonstationary as the camera is moving with the spacecraft and taking pictures of a target which may also be in motion. Hence, to gain an added insight into the problem, we explore the application of the pattern recognition theory for estimating object motion parameters based on a sequence of images. Dynamic scene analysis is receiving increasing attention from researchers in image processing and pattern recognition. Three-dimensional projection, optical flow and trajectory determination are the common approaches for determining object motion from a video sequence. A brief description of these is given below, followed by a description of the approach that is used in this work.

Three-dimensional projection techniques entail an inverse projection of the 2-dimensional (2-D) image frame onto a 3-D space. This approach makes use of the fact that all motion is in fact 3-D and consists of both translational and rotational components. A frame is a 2-D representation of the 3-D scene and may lead to ambiguity about the real scene since many different scenes could produce the same 2-D image. In other to get a correct depth model, one must consider the third dimension and estimate the translational and rotational motion parameters from the sequence of 2-D video frames. Roach and Aggarwal [5] describe such a technique for determining the movement of the objects from a sequence of images. Another well-known technique is that of optical flow. Optical flow methods represent motion in the image plane as sampled, continuous velocity fields. These are considered to be a powerful tool for dynamic scene analysis because they contain important information about the depth, structure and motion of objects. However, the techniques for determining optical flow are known to be computationally very complex. One approach for the computation of the optical flows is given by Jain in [6]. Another way of recovering 3-D parameters is based on trajectory determination for certain key points in the images. Trajectory-based methods rely on the recognition of the same set of feature points in two or more successive frames, and then utilize the correspondence between them to extract motion information. Such methods have attracted a lot of attention because they are simpler to compute than optical flow methods. Sethi and Jain [7] present an algorithm for determining trajectories of certain feature points from a sequence of images and use it to extract knowledge about the third dimension.

It is recognized that some sort of movement compensation must be accomplished in order to make the frame recovery more meaningful. Since the camera is not stationary but moves with the parent spacecraft as it approaches the target, the translation and the rotation of the target with respect to the camera can be significant, and must be estimated and accounted for in our prediction process. The accuracy of the prediction process is degraded by that portion of the picture area which can be classified as being nonstationary. Movement-compensated interframe prediction offers a promising approach to improving the accuracy of prediction by estimating and compensating for the nonstationary part of the image. In our specific application of target tracking, we are not so concerned with determining the shape of the object as with its relative orientation to the parent spacecraft. This is because the target shape would almost always be known to us a priori. Shape determination has different applications such as in computerized tomography where a physician may be interested in determining the shape and location of a tumor in a patient. For this reason, we apply movement compensation algorithms to improving the accuracy of prediction instead of the 3-D projection techniques, which usually involve solving of a complex set of nonlinear equations. In Chapter 5, we describe one of the techniques reported in the literature that we applied for the extraction of motion data and show that it is possible to improve the accuracy of prediction by compensating for motion.

We demonstrate the performance of the algorithms via a subjective evaluation of the reconstructed frames and also via some standard objective measures of performance such as mean-square error, mean- absolute error and signal-to-noise ratio. These performance measures and the motivation to use those rather than a visual evaluation alone are described in detail in Chapter 2.

To demonstrate the performance of the algorithms, we use a set of 8 frames. The frames are from a video tape of a spacecraft-to-spacecraft docking simulation, and represent the kind of data a remote teleoperator may have to work with. A typical video tape would have a frame rate of 30 frames per second. We deliberately selected frames which were not consecutive, but skipped over several frames in between. This is done so as to create a situation which is much worse than what a teleoperator would normally encounter and thereby obtain very conservative results. These data are considered relevant because the main motivation for the work is to aid in remote piloting operation. Each frame of the image data consists of 512 rows of picture elements (pixels) with 512 pixels in each row. The data are digitized with 8 bits per pixel, which is equivalent to a representation of the image pixel intensity on a scale of 0 to 255, with 255 representing the brightest intensity. The digitization results in a 512×512 array of integers. We use these digitized frames for evaluating the performance of a fixed predictor versus an adaptive predictor. The digitized data are processed on a VAX 11/750. In order to better draw any conclusions from the study, we increased the number of cases by 4 by considering 256×256 pixel subimages. Furthermore, in order to ease the computational burden imposed by the size of the matrices, especially the computation of matrix inverses, we process the images in 32×32 pixel subblocks. For obtaining the full frame, all the subblocks are pasted together in the proper order.

For a subjective evaluation of the frame estimates and comparison with the original frames, we use the COMTAL Vision One/20 system. It is a complete image processing system consisting of a fully integrated LSI-11 processor, image processing electronics and application firmware for image display. The system allows digitization of images, as well as display of digitized data, i.e., analog-to-analog (A D) and digital-to-analog (D/A) conversion. The digitized frame data were provided in the UNIX TAR (Tape Archive) format which is not compatible with the VMS operating system of the VAX 11/750 processor. In order to convert the raw binary data into real number matrices for processing on the VAX processor, we first arrange the raw data into block structured files consisting of 512 byte blocks. Subsequently, we use a set of standard tape utilities and also some special programs, to read the binary files and convert them into real number matrices for algorithm processing. The frame estimates obtained via the algorithms are likewise converted into block-structured binary files and transported to the COMTAL machine for D/A conversion. Chapter 6 gives a discussion of the simulation results. We do the processing in software to investigate the feasibility of the approach. However, in practice, the processing would be done in hardware with the use of high-speed adders and multipliers. Since this processing would be done at a ground control station instead of on board a spacecraft, we are not so constrained by the weight, power, and size of the processing equipment.

The organization of the thesis is as follows. Chapter 2 presents a formulation of the image processing problem and describes a set of objective performance criteria used for performance evaluation of the approaches. Chapter 3 presents the fixed predictor approach. The prediction is applied to both single and multiple steps of prediction using both a single frame and multiple frames. Performance of the algorithm is evaluated using the performance criteria outlined in Chapter 2. Chapter 4 presents the application of the adaptive predictor technique to the image processing problem. We also describe the peculiarities of the image processing problem and the resulting mathematical complexity involving matrix manipulations. The results are presented for a suboptimal adaptive predictor. Chapter 5 presents the approach used for displacement measurement between consecutive frames and show the effect on improving the accuracy of frame estimates. Chapter 6 presents pictures of the frame estimates derived via the various approaches along with the original pictures and discuss the results. Chapter 7 presents the conclusions of the work. Based upon our findings, some areas for future research are suggested.

CHAPTER 2

IMAGE PROCESSING PROBLEM FORMULATION

2.1. Introduction

In this chapter, we give a mathematical representation of the image processing problem stated in Chapter 1. For solving the prediction problem, we model the image processing problem as a discrete-time linear state vector model. This model is used subsequently in Chapters 3 and 4 to derive fixed and adaptive estimates of video frames. For modeling the image sequence, we represent a sequence of video frames as a discrete-time nonstationary process with each individual frame being represented as an N-dimensional vector, f(k). Since the frames are 2-dimensional and have N_1 rows of N_2 pixels each, as shown in Figure 2, therefore, f(k) can be considered to be a column vector of $(N_1 \times N_2)$ elements.

As explained in Chapter 1, in order to ease the computational complexity, we process 32×32 pixel blocks of the image instead of the entire 256×256 pixel subimages. Also, we take advantage of the adjacent-pixel correlation, both within a frame and between successive frames, to define the system matrices by representing f(k) as a 32×32 matrix instead of a 1024×1 vector. Thus, we can make use of matrix manipulation algorithms to derive the frame estimates. This statistical model assumes that each 32×32 block of pixels represents a two-dimensional separable wide-sense stationary process. In reality, however, the pixels of a block are dependent on the pixels in the neighboring blocks. This is due to the inherent non-separability of the images, and the resulting modeling error is seen as blockiness in the reconstructed images where smooth lines are expected. This can be seen in the simulation pictures presented in Chapter 6. In applications where such degradation is intolerable such as medical applications of computerized tomography, there are approaches for overcoming the border effect. One approach involves using overlapping subblocks and subsequently discarding the borders. In our application, however, this minor degradation in the reconstructed picture quality is not a problem. This is because the main aim in remote piloting is not a determination of the shape of the object, but its relative orientation to the parent

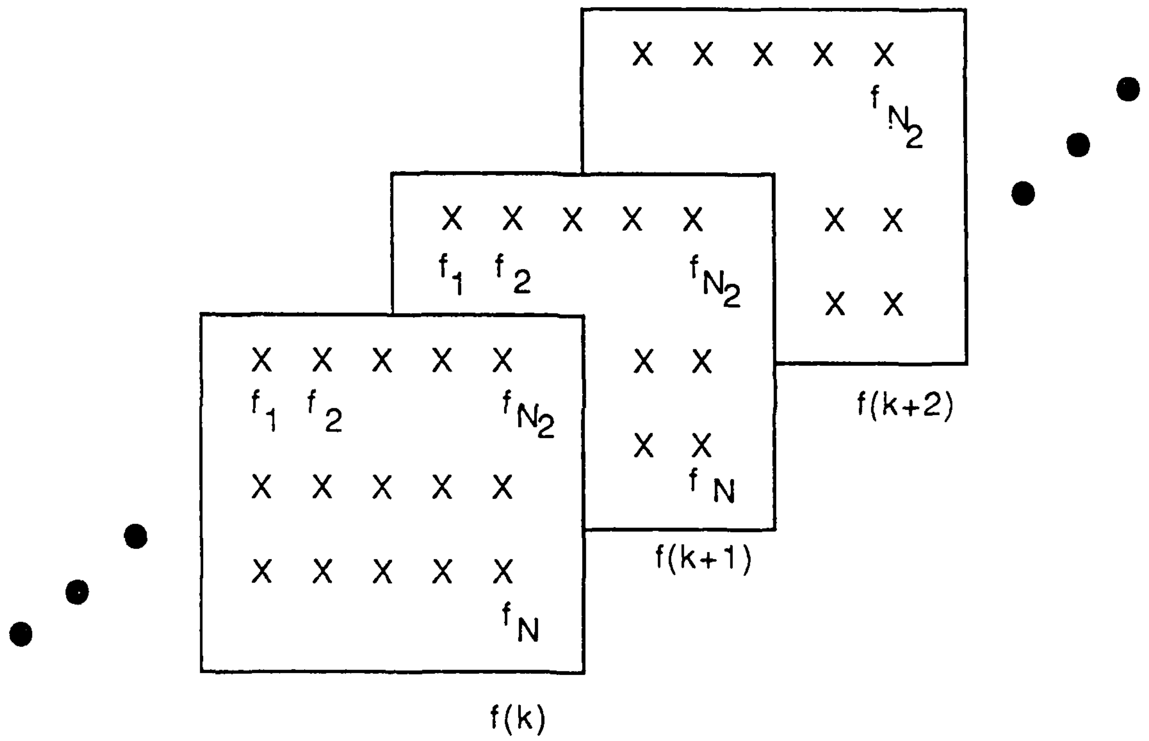


Figure 2. Representation of the video image sequence as an N-dimensional vector ($N=N_1 \times N_2$).

spacecraft. Also, the shape of the object is almost always known a priori. For these reasons, processing of the images in 32×32 pixel subblocks is considered a good compromise between picture quality and computational complexity.

What follows is a mathematical model of the image processing problem.

2.2. Equation Formulation

The source is a sequence of N-dimensional vectors $\{f(k)\}$,

$$f(k) = (f_1(k), f_2(k), \ldots, f_N(k))^T , \tag{1}$$

where N is the number of elements in vector f(k), and k is the frame number in the sequence.

For the image processing application, each vector f(k) represents a video frame with N number of pixels. For a frame containing N_1 rows and N_2 columns,

$$N = N_1 \times N_2 .$$

The scene dynamics are modeled as a state vector model where a frame is represented as a state vector. The structure of the first-order model is

$$f(k+1) = Af(k) + W(K) , \tag{2}$$

where the suffixes k and k+1 are discrete time instants. Matrix A represents system parameters, and W(k), the system modeling error.

In order to improve the accuracy of prediction, it is often helpful to increase the order of the model so as to whiten the modeling error, W. This is equivalent to estimating f(k+1) based upon not just f(k), but also f(k-1),f(k-2),....,f(k-M). In this case, the system model has the following structure

$$f(k+1) = A_1 f(k) + A_2 f(k-1) + \cdots + A_{M+1} f(k-M) + W(k) . \tag{3}$$

Equation (3) is readily recognized as a higher-order state model where now the state vector is

$$[f(k)^T \, f(k-1)^T \, \cdots \, f(k-M)^T]^T \, . \tag{4}$$

The observed data is corrupted by channel noise representing the digital data transmission error associated with the satellite link. The general system model for a first-order model is as follows:

$$f(k+1) = A(k)\, f(k) + W(k) \; ; \quad f(k) \in R^N \tag{5}$$

where A (k) is the system matrix,

f(k) is the N-dimensional state vector,

W(k) is the modeling error between the actual value f(k+1) and the predicted value ftilde(k+1/k),

$\tilde{f}(k+1/k)$ is the estimate at time k+1 knowing measurements at time instants up to and including k.

This state vector model is similar to the model used in Kalman filter application. However, here we are not using any observations for updating the estimates at each instant as the new data becomes available. Instead, we consider a subset of the available frames to predict the next few frames using the state vector equation only, and show that as more frames are received, we can derive a better estimate due to a decrease in the number of steps of prediction required.

A prediction of the vector $f^{\sim}(k+1)$ is formed based on the past statistics, f(k),f(k-1),....,f(k-M), using either the fixed or the adaptive predictor described in Chapters 3 and 4.

The prediction, $f^{\sim}(k+1)$, is obtained from

1. f(k)

   (one-step prediction, $\tilde{f}(k+1)/f(k)$), or

2. f(k) and f(k-1)

   (two-step prediction, $(f^{\sim}(k+1)/f(k), f(k-1))$, or

3. f(k), f(k-1), f(k-2),.... and f(k-M)

   (multi-step prediction, $(f^{\sim}(k+1)/f(k), f(k-1),....,f(k-M))$.

The model presented above is used in Chapters 3 and 4 to derive frame estimates via fixed and adaptive predictors. The estimate, $\tilde{f}(k+1)$, is compared with the actual frame $f(k+1)$ to obtain both a subjective and an objective evaluation. For the subjective evaluation, the frame data is converted from digital-to-analog format as described in Chapter 1 and is displayed on a computer monitor. The performance is also analyzed using a set of objective measures of performance which are described in detail in the following section.

2.3. Performance Criteria

Having modeled our system, our next objective is to outline a set of criteria for analyzing and comparing the performance of various techniques for image data prediction. In this section, we discuss criteria for an objective evaluation of the reproduced images. It is recognized that the visual fidelity assessment of reconstructed video images is based upon a subjective evaluation of the images. This is because the ultimate user of these images is man. Seyler [8] describes visual communications and the psychophysics of human vision and suggests that the objective of television is to produce "as accurately as practicable a realistic replica of the natural environment shown, i.e., (to create) in the viewer's mind the illusion of direct communication." We assume that the television cameras employed on board the spacecraft conform to an accepted standard and regard the original video frames as an accurate replica of the real scene. Our objective then is to reproduce those images with as little distortion as possible. Since the ultimate destination of these images is a man-in-the-loop, the most important criterion is his accurate perception of the scene.

However, for the purpose of designing communications systems and for comparing performances of alternative systems and designs, one also requires an explicit evaluation of the reproduced images. It is widely recognized that to mathematically model man's sense of vision, including luminance and chrominance vision, is a very complex problem. It is an accepted practice to employ measurements which are analytically more tractable then the mathematical models of human vision and have criterion values. This applies both to analog and digital transmissions, and monochrome and color images. In this work, however, we are concerned with digital transmission and monochrome images only.

In employing objective measures of performance for assessing visual fidelity of reconstructed images, it is implied that video distortion is identifiable with errors in reproduction and will result in a poor performance with respect to the objective criteria employed. Otherwise, the criteria would have only a limited value or none at all. A number of papers on image processing have addressed this issue and sought to find a numerically-valued measure of distortion which has a reasonable correspondence with the subjective evaluation by a human interpreter. Hall, Budrikis, and Mannos [9,10,11] address this problem and suggest some alternatives.

In this work, since we also propose to use subjective evaluation for the reproduced images, we have restricted ourselves to commonly used objective measures of performance which are described below.

2.3.1. Standard image quality measures

Some commonly used image quality measures are defined below.

1. Mean-Square Error

One of the most commonly used quality or distortion measure is mean-square error (MSE). MSE is defined as

$$MSE = E\,[\tilde{f}(k) - f(k)]^2 ,$$

where $F(k)$ is the estimate of the frame and $f(k)$, the actual frame .

For an $N \times N$ discrete image, MSE may be defined as

$$MSE = \frac{1}{N \times N} \sum_{i=1}^{N} \sum_{j=1}^{N} [\tilde{f}(i,j) - f(i,j)]^2 . \qquad (6)$$

This measure is attractive because it is analytically tractable. Its limitation is that on certain kinds of images, it does not correspond very closely with human evaluation.

2. Normalized Mean-Square Error

One can also define an image quality measure based on MSE and energy normalization. It is called normalized mean square error (NMSE), and it is defined as

$$\text{NMSE} = \frac{\text{MSE between the original and reconstructed frame}}{\text{variance of the original image}}$$

For an N×N discrete image,

$$\text{NMSE} = \frac{\sum_{i=1}^{N} \sum_{j=1}^{N} [\tilde{f}(i,j) - f(i,j)]^2}{\sum_{i=1}^{N} \sum_{j=1}^{N} [f(i,j)]^2} . \tag{7}$$

The NMSE measure performs better than the MSE. It maintains the analytical tractability of the MSE and is equally simple to compute.

3. Mean Absolute Error

Another simple measure is that of mean absolute error (MABSE). Its appeal is mainly due to the simplicity. It is defined as

$$\text{MABSE} = \frac{1}{N \times N} \sum_{i=1}^{N} \sum_{j=1}^{N} | \tilde{f}(i,j) - f(i,j) | , \tag{8}$$

where $|\cdot|$ represents the absolute value of the argument. The MABSE performs well at low intensity levels since incremental values at low intensity levels are more noticeable than those at high intensity values.

4. Signal-to-Noise Ratio

The peak-to-peak signal-to-noise ratio (SNR) is defined as

$$\text{SNR} = 10 \log \frac{(\text{peak-to-peak signal value})^2}{\text{MSE}} , \tag{9}$$

where peak-to-peak signal value is 255 for an image quantized with 8 bits. The SNR is a commonly accepted image quality measure and has a reasonably good correlation with the distortion in the reproduced image.

The results in Chapters 3, 4, and 5 are tabulated according to these criteria for various cases.

CHAPTER 3

APPLICATION OF A FIXED PREDICTOR FOR IMAGE DATA PREDICTION

3.1. Introduction

In Chapter 3, we present a mathematical model that could be used to describe the dynamical behavior of the image processing problem. The problem with modeling the scene dynamics as a state vector model is a complex one since the system matrix, A, of Equation (5) is highly scene-dependent, and also depends on how rapidly the object is moving. In an application such as video image processing, where the system and noise models are either ill-defined or not completely specified, it is feasible to estimate a model using certain properties which are peculiar to video images. For developing a fixed predictor, we derive a state vector model by exploiting the high level of adjacent-pixel correlation inherent in video images. This is true both of adjacent pixels of a frame as well as corresponding pixels of adjacent frames. We call this a fixed predictor because we use a fixed system matrix, A. This predictor is then applied to the most recent set of video frames received for the purpose of estimating the next few frames of the sequence.

The advantages of using this predictor for image data prediction are as follows:

(1) To carry out on-line prediction of image data using frames as they become available instead of doing frame refresh, which depends only on the last frame received. Here, we utilize more data to try to derive a more accurate estimate.

(2) To incorporate modeling error as system noise to improve the accuracy of the prediction even more. This is not feasible with frame refresh alone.

(3) To derive interframe motion using a set of the last frames received via a determination of pixel trajectories or other techniques and use the displaced frames instead of the actual ones for the prediction. Thus, we can do motion compensation which the technique of frame refresh does not allow.

In Chapter 6, we present the pictures of the frame estimates derived via various combinations of available frames for both single as well as multiple steps of prediction. The performance according to the criteria outlined in Chapter 2 is summarized in Tables 1-5. The results obtained with the use of frame refresh only are summarized in Tables 6 and 7 for a comparison. It is shown that the use of the fixed predictor provides better criterion values (MSE and SNR) than the frame refresh. However, an evaluation of both the objective and the subjective merit of the reconstructed frames seems to indicate that there is not a tremendous improvement over frame refresh.

3.2. Equation Formulation

We use a sequence of frames $\{f(k)\}$, where each $f(x)$ is an $N_1 \times N_2$- dimensional matrix. As explained in Chapter 2, $f(k)$ is considered to be a 32×32 matrix. The structure of the first-order model is as given in Chapter 2, Equations (2)-(5). The unmodeled dynamics are accounted for by a model error term. Just as a reference, the general system equation for a first-order model is as follows:

$$f(k+1) = A(k)\, f(k) + W(k)\, ; \quad f(k) \in R^N \tag{10}$$

where $A(k)$ is the system matrix

$f(k)$ is the N-dimensional ($N = N_1 \times N_2$) state vector,

$W(k)$ is the modeling error between the actual value $f(k+1)$ and the predicted value $\tilde{f}(k+1/k)$,

$\tilde{f}(k+1/k)$ is the estimate of $f(k+1)$ knowing measurements at time instants up to and including k, and

the suffixes, k and k+1, are discrete time instants.

Assumption The statistical properties of $W(k)$ are assumed to be zero-mean white Gaussian noise with the covariance given by

$$E\{W(k)\, W(l)^T\} = Q(k)\delta_{kl}$$

Table 1. Fixed Predictor, Next-Frame Prediction

$$\tilde{f}(K+1) = \Phi(K+1,K)*f(K)$$

DESCRIPTION	BLK #	MABSE	MSE	% NMSE	SNR (dB)	OVERALL IMAGE SNR (dB)
IMAGE 2 FROM IMAGE 1	1	3.01	18.6	0.16	35.4	37.4
	2	1.90	8.8	0.08	38.7	
	3	1.9	14.2	0.15	36.6	
	4	1.7	5.5	0.05	40.8	
IMAGE 3 FROM IMAGE 2	1	4.8	40.0	0.34	32.1	33.6
	2	2.5	15.1	0.14	36.3	
	3	4.96	48.0	0.47	31.3	
	4	2.5	11.3	0.11	37.6	
IMAGE 4 FROM IMAGE 3	1	5.5	54.0	0.47	30.8	33.3
	2	2.5	14.6	0.14	36.5	
	3	4.7	42.4	0.40	31.9	
	4	2.22	9.55	0.09	38.3	
IMAGE 5 FROM IMAGE 4	1	5.86	57.7	0.49	30.5	33.3
	2	2.74	16.7	0.16	35.9	
	3	4.35	37.5	0.35	32.4	
	4	2.3	10.0	0.09	38.1	
IMAGE 6 FROM IMAGE 5	1	9.11	166.0	1.45	25.9	27.1
	2	9.35	277.0	1.80	24.6	
	3	5.29	55.8	0.52	30.7	
	4	5.03	54.2	0.47	30.8	
IMAGE 7 FROM IMAGE 6	1	5.88	71.9	0.62	29.6	30.0
	2	6.62	121.0	0.92	27.3	
	3	4.7	46.0	0.43	31.5	
	4	3.1	20.3	0.17	35.1	
IMAGE 8 FROM IMAGE 7	1	6.8	87.4	0.72	28.7	28.4
	2	8.7	193.0	1.4	25.3	
	3	5.92	69.3	0.60	29.7	
	4	3.42	25.2	0.21	34.1	

Table 2. Fixed Predictor, Two-Frame Ahead Prediction

$$\tilde{f}(K+2) = \Phi(K+2,K)*f(K)$$

DESCRIPTION	BLK #	MABSE	MSE	% NMSE	SNR (dB)	OVERALL IMAGE SNR (dB)
IMAGE 3 FROM IMAGE 1	1	4.93	41.2	0.35	32.0	33.1
	2	2.65	15.0	0.14	36.4	
	3	5.82	57.0	0.49	30.6	
	4	2.66	13.3	0.13	36.9	
IMAGE 4 FROM IMAGE 2	1	4.80	41.4	0.36	32.0	31.7
	2	2.37	11.2	0.11	37.6	
	3	8.65	108.0	0.88	27.8	
	4	3.04	17.1	0.16	35.8	
IMAGE 5 FROM IMAGE 3	1	5.84	57.5	0.49	30.5	32.3
	2	2.87	18.2	0.17	35.5	
	3	6.38	63.6	0.52	30.1	
	4	2.88	15.8	0.15	36.1	
IMAGE 6 FROM IMAGE 4	1	7.95	135.0	1.18	26.8	27.1
	2	10.2	240.0	1.91	24.3	
	3	6.08	62.9	0.52	30.2	
	4	5.81	67.8	0.58	29.8	
IMAGE 7 FROM IMAGE 5	1	9.04	148.0	1.27	26.4	26.1
	2	12.2	343.0	2.62	22.8	
	3	6.89	76.9	0.62	29.3	
	4	5.65	68.4	0.59	29.8	
IMAGE 8 FROM IMAGE 6	1	7.97	109.0	0.89	27.8	27.2
	2	10.4	254.0	1.84	24.1	
	3	7.84	103.0	0.79	28.0	
	4	3.65	30.9	0.26	33.2	

Table 3. Fixed Predictor Three-Frame Ahead Prediction

$$\tilde{f}(K+3) = \Phi(K+3,K)*f(K)$$

DESCRIPTION	BLK #	MABSE	MSE	% NMSE	SNR (dB)	OVERALL IMAGE SNR (dB)
IMAGE 4 FROM IMAGE 1	1	5.18	46.7	0.40	31.4	31.9
	2	2.72	14.8	0.14	36.4	
	3	7.68	89.4	0.73	28.6	
	4	2.96	15.7	0.15	36.2	
IMAGE 5 FROM IMAGE 2	1	5.27	50.1	0.43	31.1	31.3
	2	2.79	17.2	0.16	35.8	
	3	8.52	104.0	0.84	28.0	
	4	3.35	21.2	0.20	34.9	
IMAGE 6 FROM IMAGE 3	1	8.94	154.0	1.34	26.3	27.1
	2	9.67	224.0	1.78	24.6	
	3	5.61	61.0	0.50	30.3	
	4	6.01	72.8	0.63	29.5	
IMAGE 7 FROM IMAGE 4	1	8.41	128.0	1.1	27.1	26.5
	2	12.0	328.0	2.5	23.0	
	3	6.11	61.4	0.5	30.2	
	4	5.83	67.9	0.6	29.8	
IMAGE 8 FROM IMAGE 5	1	9.9	179.0	1.47	25.6	24.9
	2	14.8	478.0	3.45	21.3	
	3	8.4	108.0	0.83	27.8	
	4	6.1	74.5	0.63	29.4	

Table 4. Fixed Predictor, Next-Frame Prediction Using a Last Two Frames Received

$$\tilde{f}(K+1) = A_1 * f(K) + A_2 * f(K-1)$$

DESCRIPTION	BLK #	MABSE	MSE	% NMSE	SNR (dB)	OVERALL IMAGE SNR (dB)
IMAGE 3 FROM IMAGE 2 & IMAGE 1	1	4.63	36.7	0.31	32.5	33.7
	2	2.42	13.3	0.13	36.9	
	3	5.45	51.2	0.44	31.0	
	4	2.4	10.8	0.10	37.8	
IMAGE 4 FROM IMAGE 3 & IMAGE 2	1	4.68	39.9	0.34	32.1	33.4
	2	2.19	9.75	0.09	38.2	
	3	6.16	60.1	0.49	30.3	
	4	2.29	10.1	0.09	38.1	
IMAGE 5 FROM IMAGE 4 & IMAGE 3	1	5.28	45.8	0.39	31.5	33.7
	2	2.52	13.9	0.13	36.7	
	3	4.96	41.1	0.33	32.0	
	4	2.31	10.2	0.09	38.1	
IMAGE 6 FROM IMAGE 5 & IMAGE 4	1	8.13	139.0	1.21	26.7	27.4
	2	9.43	224.0	1.78	24.6	
	3	5.52	54.5	0.45	30.8	
	4	5.16	55.9	0.48	30.7	
IMAGE 7 FROM IMAGE 6 & IMAGE 5	1	6.13	69.9	0.60	29.7	29.1
	2	8.03	169.0	1.29	25.8	
	3	5.52	51.3	0.42	31.0	
	4	3.72	28.8	0.25	33.5	
IMAGE 8 FROM IMAGE 7 & IMAGE 6	1	6.83	80.1	0.66	29.1	28.4
	2	8.84	193.0	1.4	25.3	
	3	6.78	77.1	0.59	29.3	
	4	3.26	23.4	0.198	34.4	

Table 5. Fixed Predictor, Next-Frame Prediction
Using Last Three Frames Received

$$f(K+1) = A_1*f(K)+A_2*f(K-1)+A_3*f(K-2)$$

DESCRIPTION	BLK #	MABSE	MSE	% NMSE	SNR (dB)	OVERALL IMAGE SNR (dB)
IMAGE 4 FROM IMAGE 3 & IMAGE 2 & IMAGE 1	1	4.63	38.8	0.34	32.2	33.1
	2	2.20	9.79	0.09	38.2	
	3	6.55	66.7	0.55	29.9	
	4	2.40	10.9	0.10	37.7	
IMAGE 5 FROM IMAGE 4 & IMAGE 3 & IMAGE 2	1	4.96	42.1	0.36	31.9	33.5
	2	2.43	12.7	0.12	37.1	
	3	5.58	48.9	0.40	31.2	
	4	2.44	11.6	0.11	37.5	
IMAGE 6 FROM IMAGE 5 & IMAGE 4 & IMAGE 3	1	8.15	135.0	1.18	26.8	27.5
	2	9.44	222.0	1.77	24.7	
	3	5.11	48.8	0.40	31.2	
	4	5.37	60.0	0.52	30.3	
IMAGE 7 FROM IMAGE 6 & IMAGE 5 & IMAGE 4	1	6.42	77.2	0.66	29.3	28.5
	2	9.11	207.0	1.58	25.0	
	3	5.35	47.7	0.39	31.3	
	4	4.23	37.2	0.319	32.4	
IMAGE 8 FROM IMAGE 7 & IMAGE 6 & IMAGE 5	1	6.89	82.8	0.68	29.0	27.9
	2	9.89	233.0	1.69	24.5	
	3	6.91	76.6	0.58	29.3	
	4	3.62	29.1	0.25	33.5	

Table 6. Next-Frame Prediction by Frame Refresh

DESCRIPTION	BLK #	MABSE	MSE	% NMSE	SNR (dB)	OVERALL IMAGE SNR (dB)
IMAGE 2 FROM IMAGE 1	1	3.37	22.5	0.19	34.9	36.2
	2	2.14	8.8	0.08	38.7	
	3	3.35	22.6	0.19	34.6	
	4	2.16	8.94	0.08	38.6	
IMAGE 3 FROM IMAGE 2	1	5.07	44.2	0.37	31.7	33.2
	2	2.72	18.2	0.17	35.5	
	3	5.04	44.8	0.38	31.6	
	4	2.76	16.8	0.16	35.9	
IMAGE 4 FROM IMAGE 3	1	5.73	59.1	0.51	30.4	32.3
	2	2.70	15.7	0.15	36.2	
	3	5.87	62.4	0.54	30.8	
	4	2.76	16.6	0.16	35.9	
IMAGE 5 FROM IMAGE 4	1	6.07	61.9	0.53	30.2	32.1
	2	2.90	17.8	0.17	35.6	
	3	6.22	65.1	0.56	30.0	
	4	2.9416.3	0.15	36.0		
IMAGE 6 FROM IMAGE 5	1	9.23	171.0	1.49	25.8	25.0
	2	9.42	226.0	1.80	24.6	
	3	9.75	189.0	1.65	25.4	
	4	10.0	249.0	1.96	24.2	
IMAGE 7 FROM IMAGE 6	1	6.09	76.2	0.66	29.3	27.8
	2	6.85	132.0	1.00	26.9	
	3	6.10	78.4	0.68	29.2	
	4	7.49	147.0	1.10	26.5	
IMAGE 8 FROM IMAGE 7	1	7.01	94.2	0.77	28.4	
	2	9.04	206.0	1.49	25.0	
	3	6.95	95.0	0.78	_26.2 28.4	
	4	9.92	233.0	1.64	24.5	

Table 7. Frame Refresh, Two-Frame Ahead Prediction

$$\tilde{f}(K+1) = A_1 * f(K) + A_2 * f(K-1) + A_3 * f(K-2)$$

DESCRIPTION	BLK #	MABSE	MSE	% NMSE	SNR (dB)	OVERALL IMAGE SNR (dB)
IMAGE 3 FROM IMAGE 1	1	5.11	44.6	0.38	31.6	33.4
	2	2.68	16.0	0.15	36.1	
	3	5.08	45.3	0.38	31.6	
	4	2.70	14.4	0.14	36.6	
IMAGE 4 FROM IMAGE 2	1	5.14	47.2	0.41	31.4	33.4
	2	2.54	13.2	0.13	36.9	
	3	5.19	48.4	0.42	31.3	
	4	2.58	11.6	0.11	37.5	
IMAGE 5 FROM IMAGE 3	1	6.11	63.9	0.54	30.1	32.0
	2	2.91	17.6	0.17	35.7	
	3	6.18	66.5	0.57	29.9	
	4	2.93	15.9	0.15	36.1	
IMAGE 6 FROM IMAGE 4	1	8.26	141.0	1.23	26.6	25.3
	2	9.74	228.0	1.81	24.6	
	3	8.59	153.0	1.34	26.3	
	4	10.4	249.0	1.93	24.2	
IMAGE 7 FROM IMAGE 5	1	9.31	155.0	1.33	26.2	24.1
	2	11.8	330.0	2.52	22.9	
	3	9.64	166.0	1.43	25.9	
	4	12.7	366.0	2.74	22.5	
IMAGE 8 FROM IMAGE 6	1	8.0	111.0	0.91	27.7	25.3
	2	10.4	259.0	1.87	24.0	
	3	7.73	106.0	0.87	27.9	
	4	11.5	292.0	2.06	23.5	

where $Q(k)$ is a positive semi-definite matrix and δ_{kl} is the Kronecker delta

$$= \begin{cases} 1 & \text{for } k = l \\ 0 & \text{for } k = l \end{cases}.$$

Algorithm for fixed prediction in the absence of modeling error.

(1) Relation between prediction $\tilde{f}(k+1)$ and $f(k)$.

$$\tilde{f}(k+1/k) = A(k)\, f(k)\,. \tag{11}$$

(2) For multiple steps of prediction, the relation between $f(k+M/k)$ and $f(k)$ is given by

$$\tilde{f}(k+M/k) = \Phi(k+M,k)\, f(k) \tag{12}$$

where Φ is the state transition matrix.

$$\begin{aligned} \Phi(m,m) &= I \\ \Phi(m,n) &= A(m-1)\,.\,A(m-2) \cdots A(n) \end{aligned} \tag{13}$$

A generalized state model is

$$f(k) = \Phi(k,k-M)\, f(k-M) + W(k)\,. \tag{14}$$

It is seen that the fixed predictor resembles the standard estimation technique of Kalman filter with one exception. Here, we are not using a measurement vector to update the parameters between samples. However, the fixed predictor allows us to update the state vector by using the most recent set of frames, and thus provides a better estimate as the number of steps of prediction is lowered.

3.3. Performance Evaluation

In order to find a suitable system matrix A, we used the fact that video images are characterized by very high pixel correlation, both in the space domain and the time domain, i.e., in the temporal direction. We assume that the pixels are zero-mean samples of a separable Markov process. Then, by assuming a fixed adjacent-pixel correlation, the structure of the A matrix is as shown in Figure 3. This correlation structure is used to describe both spatial and temporal correlation, i.e., between adjacent pixels of a frame as well as between corresponding pixels of adjacent frames. We assume a fixed pixel correlation of 0.96 for both intraframe and interframe correlation. This results in an A matrix shown in Appendix B for a first-order system for one frame-ahead prediction using the last frame received. Other matrices are selected likewise to increase the system order for increasing the accuracy of prediction.

Tables 1-5 show the results of applying this algorithm to obtain frame estimates for the next frame (Table 1), two-frame ahead prediction (Table 2), three-frame ahead prediction (Table 3), next-frame prediction using the last two frames received (Table 4), and using the last three frames (Table 5). The results of the frame refresh technique applied to the next-frame prediction are summarized in Tables 6 and 7 for a prediction based on the last frame received, and the last two frames, respectively. These are presented for a contrast with the fixed predictor results. It is seen from these tables that the fixed predictor provides better criterion values than frame refresh in almost all the cases. The results in Tables 1-5 were obtained without compensating for motion compensation. It is shown in Chapter 5 that it is possible to improve upon the performance of the predictor by estimating and compensating for the interframe displacements. The pictures of the frame estimates are presented in Chapter 6. From a subjective evaluation of the reconstructed frames, it appears that there is not a tremendous advantage to using a fixed predictor over a simple frame refresh in spite of what the criterion values indicate. All these results and a subjective evaluation of the reconstructed frames are discussed in detail in Chapter 6.

$$\begin{bmatrix} 1 & \rho & \rho^2 & \rho^3 & . & . \\ \rho & 1 & \rho & \rho^2 & . & . \\ \rho^2 & \rho & 1 & \rho & \rho^2 & . \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ . & . & \rho^3 & \rho^2 & \rho & 1 \end{bmatrix}$$

Figure [illegible] image frame

CHAPTER 4

APPLICATION OF AN ADAPTIVE PREDICTOR FOR IMAGE DATA PREDICTION

4.1. Introduction

In the case of a fixed predictor, we assume a certain model for describing the interframe relationship of a video sequence. That model is then used for prediction of frames using a set of past frames. In the case of an adaptive predictor, we estimate the state vector model using the interframe correlation of the available frames, and by making an assumption that the model can be approximated as a wide-sense stationary random process. In an application such as remotely piloted spacecraft, it appears to be a reasonable assumption especially in the docking mode which is characterized by very slow motion.

The approach used for developing the adaptive predictor is the classical parameter estimation technique of generalized least squares. We seek to find an optimal adaptive predictor which would provide a least mean-square solution to the prediction problem, i.e., minimize the square of the error between the original and the reconstructed frames. It is shown in this chapter that the ill-conditioned nature of the image processing problem, specifically the sample covariances, render it computationally a very complex problem. It is shown that when we represent the images by the intensity values of the pixels, the resulting matrices are almost always singular or nearly singular. In the image processing problem, the image degradation is represented as a transformation. Hence, to recover the original image from a degraded one or to reconstruct images often requires computation of inverse transformations, which is mathematically represented as a matrix inversion problem. It is shown that such matrices have zero or near-zero eigenvalues and thus it is not possible to find an inverse. We show that the well-known technique of adding a disturbance along the diagonal to stabilize singular matrices does not overcome the problem of the singularity of these matrices. We then explore the use of Singular Value Decomposition (SVD) in order to isolate and discard the near zero singular values and attempt to find an inverse by effectively reducing the rank of the matrix. It is shown that the large variation in the singular values effectively amplifies the noise and makes the recovered image unacceptable in terms of visual fidelity. Because of the mathematical complexity associated with the optimal adaptive predictor, we use a suboptimal adaptive predictor which still uses the previous samples to derive the frame estimates. The intent is to evaluate the performance of this suboptimal predictor relative to that of the fixed predictor and frame refresh.

4.2. Equation Formulation

As mentioned above, the algorithm for the adaptive preditor is based on the standard technique of generalized least squares parameter estimation. The equations summarized below can be found in any standard textbook or reference in estimation theory and in [3]. A prediction of vector $\tilde{f}(k)$, namely, $\tilde{f}(k)$, is formed based on previously reconstructed vector $f\hat{}(k-1)$ as shown in Figure 4.

$$\tilde{f}(k) = B(k)\,\hat{f}(k-1)\,, \tag{15}$$

where B(k) is a time-varying predictor and f(k) is the kth frame of the sequence. The prediction can also be based upon a set of previously reconstructed vectors, $\{\hat{f}(k-\hat{1}), f(k-\hat{2}), \ldots, f(k-M)\}$. For a non-time-varying predictor,

$$\tilde{f}(k) = B\hat{f}(k-1)\,. \tag{16}$$

The B vector can also be chosen to be a diagonal matrix,

$$B = \mathrm{diag}(b_1, b_2, \ldots, b_N)^T\,, \tag{17}$$

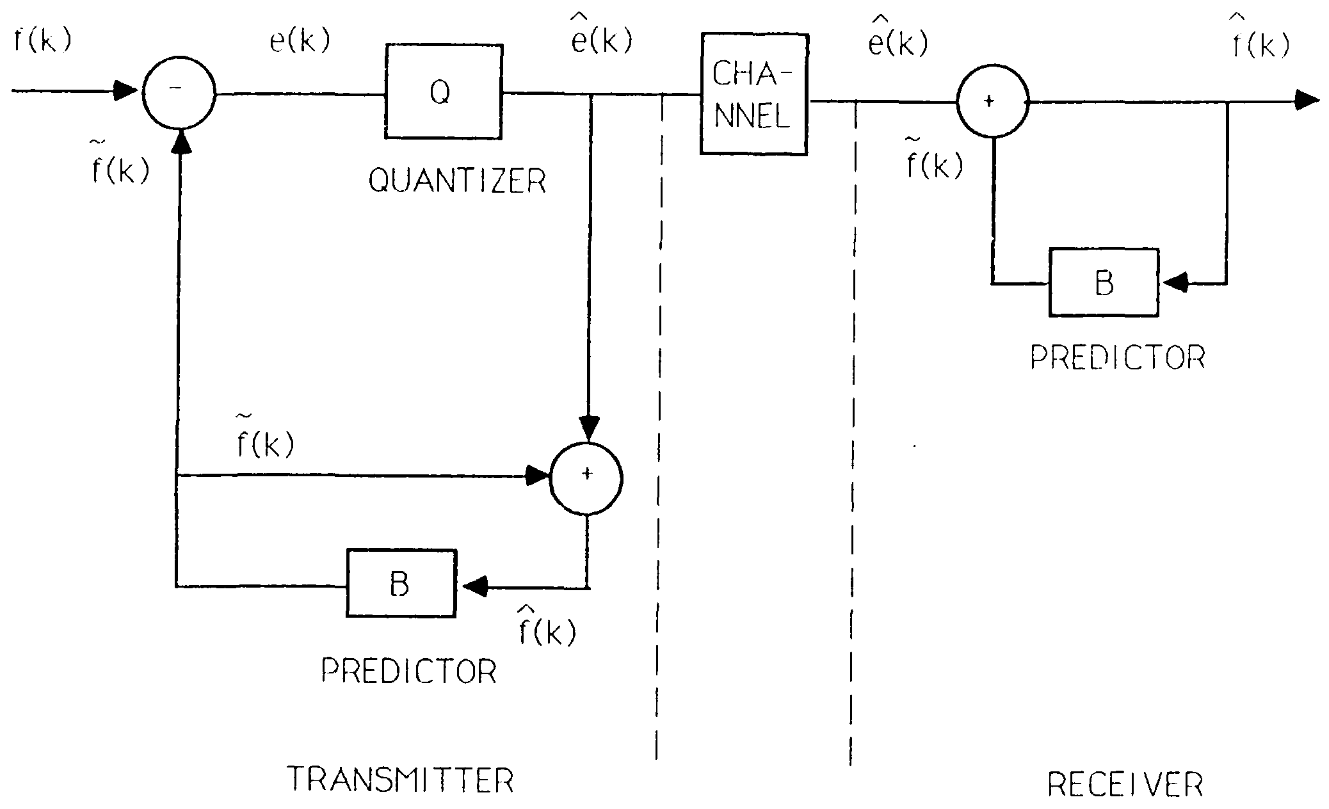
in which case each element of the estimated vector $\tilde{f}_i(k)$ depends only on the corresponding element of the previous reconstructed vector, $\hat{f}(k-1)$, i.e.,

$$\tilde{f}_i(k) = b_i^T \hat{f}(k-1)\,.$$

Using M previously reconstructed vectors,

$$\tilde{f}(k) = B(k)\,g(k)\,, \tag{18}$$

where g(k) is a column vector consisting of the M previous vectors



f(k) = estimate

f(k) = the reconstructed vector

$\tilde{f}(k)$ = the original vector

Figure 4. Feedback predictor for image sequence reconstruction.

$$g(k)^T = (\hat{f}(k-1)^T \, \hat{f}(k-2)^T \cdots \hat{f}(k-M)^T) . \tag{19}$$

Equation (18) is a generalization of Equation (15) using multiple steps of prediction.

The prediction error e(k) is

$$e(k) = \tilde{f}(k) - B(k)\, g(k) . \tag{20}$$

In the matrix form, e(k) can be expressed as

$$e(k) = \begin{bmatrix} \tilde{f}_1(k) \\ \tilde{f}_2(k) \\ \cdot \\ \tilde{f}_N(k) \end{bmatrix} - \begin{bmatrix} b_1^T(k) \\ b_2^T(k) \\ \cdot \\ b_N^T(k) \end{bmatrix} \begin{bmatrix} f_1(k-1) \\ f_2(k-1) \\ \cdot \\ f_N(k-1) \\ \cdot \\ \cdot \\ f_1(k-M) \\ f_2(k-M) \\ \cdot \\ f_N(k-M) \end{bmatrix} . \tag{21}$$

Here, we are assuming that the quantization error is negligible, i.e.,

$$\hat{f}(k) = f(k) .$$

The predictor vector B(k) consists of N vectors $b_i(k)$ corresponding to each element of the vector f(k), i.e.,

$$B(k) = (b_1(k)\, b_2(k) \cdots b_N(k))^T . \tag{22}$$

where each $b_i(k)$ consists of N×N coefficients for the i-th element of f(k). The error vector e(k) is given by

$$e_i(k) = \tilde{f}_i(k) - b_i(k)^T g(k) , \quad \text{for } i = 1,2,\ldots,N . \tag{23}$$

At this point, we select the standard parameter estimation technique of least mean squares (LMS) for developing an optimal predictor. In other words, an optimal predictor is one that minimizes the reconstruction error, e(k). Using this criterion, the optimal estimate would be the LMS estimate.

The LMS estimate f(k) is obtained by choosing the predictor matrix B(K) such that the mean square reconstruction error is minimized, i.e.,

$$MSE = E\{e_i(k)^2\} \text{ is minimized for all } i = 1,2,\ldots,N . \tag{24}$$

The error covariance matrix $R_e(k)$ is defined as

$$R_e(k) \overset{\Delta}{=} E\{e(k)\, e(k)^T\} \tag{25}$$

$$\begin{aligned} \text{Min } \{e_i(k)^2\} &= \text{Min}\{\text{Diagonal elements of } R_e(k)\} \\ &= \text{Min}\{\text{tr } R_e(k)\} . \end{aligned} \tag{26}$$

The optimal predictor B(K) minimizes the trace of $R_e(k)$ for an LMS criterion,

$$\nabla_{B(k)}\{\text{tr } R_e(K)\} = 0 . \tag{27}$$

Using simple matrix manipulations and the following properties of a trace operator,

$$\text{tr}\,(A+B) = \text{tr } A + \text{tr } B \text{ , and}$$

$$\text{tr } A^T = \text{tr } A ,$$

$$\begin{aligned} \text{tr}\{R_e(k)\} &= \text{tr } E\{e(k)\, e(k)^T\} \qquad (28) \\ &= \text{tr } E\{[f(k)-B(k)\, g(k)][f(k)^T - g(k)^T\, B(k)^T]\} \\ &= \text{tr}\{R(k,k) - B(k)\, R_{gf}(k) - R_{gf}(k)^T B(k)^T + B(k) R_g(k) B(k)^T\} \\ &= \text{tr}(R(k,k)) - 2\, \text{tr}(B(k) R_{gf}(k)) \qquad (29) \\ &\quad + \text{tr}(B(k) R_g(k) B(k)^T) , \end{aligned}$$

where the following definitions for covariance matrices apply

$$\begin{aligned} R(k_1,k_2) &\overset{\Delta}{=} E\{f(k_1)\, f(k_2)^T\} \\ R_g(k) &\overset{\Delta}{=} E\{g(k)\, g(k)^T\} \\ R_{gf}(k) &\overset{\Delta}{=} E\{g(k)\, f(k)^T\} . \end{aligned} \tag{30}$$

The optimal predictor B(k), namely, B^+ is such that it minimizes $\text{tr}\{R_e(k)\}$, i.e., from Equations (29) and (18)

$B^{+} = B(k)$ such that

$$\nabla_{B(k)}[\mathrm{tr}\, R(k)\} - 2\,\mathrm{tr}\{B(k)R_{gf}(k)\} + \mathrm{tr}\{B(k)R_g(k)B(k)^T\}] = 0 . \tag{31}$$

Using simple matrix manipulation and the following rules of matrix gradient erations,

$$\nabla_B\, \mathrm{tr}\,(BA) = A^T \quad \text{and} \tag{32}$$

$\nabla_B\, \mathrm{tr}\,(BAB^T) = BA^T + BA$, for the optimal predictor ,

$$\nabla_{B(k)}\, \mathrm{tr}\,\{R_e(k)\} = -2\, R_{gf}(k)^T + B(k)(R_g(k) + R_g(k)^T) = 0 . \tag{33}$$

Since $R_g(k)$ is symmetric, assuming that its inverse exists, the LMS predictor,

$$B^{+}(k) = R_{gf}(k)^T\, R_g(k)^{-1} . \tag{34}$$

The LMS prediction error,

$$e^{\dagger}(k) = \tilde{f}(k) - B^{\dagger}(k)\, g(k) \tag{35}$$

together with $R_{gf}(k)$ and $R_g(k)$, can be approximated using the covariance of the original data.

$$R_{gf}(k) = \begin{bmatrix} E\{f(k-1)f(k)^T\} \\ E\{f(k-2)f(k)^T\} \\ \cdot \\ E\{f(k-M)f(k)^T\} \end{bmatrix} = \begin{bmatrix} R(k-1,k) \\ R(k-2,k) \\ \cdot \\ R(k-M,k) \end{bmatrix} \quad \text{and}$$

$$R_g(k) = \begin{bmatrix} E\{f(k-1)f(k-1)^T\} & \cdots & E\{f(k-1)f(k-M)^T\} \\ \cdot & & \cdot \\ E\{f(k-M)f(k-1)^T\} & \cdots & E\{f(k-M)f(k-M)^T\} \end{bmatrix} \tag{36}$$

$$= \begin{bmatrix} R(k-1,k-1) & \cdots & R(k-1,k-M) \\ \cdot & & \cdot \\ R(K-M,K-1) & \cdots & R(K-M,K-M) \end{bmatrix} .$$

For a single-step prediction,

$$R_{gf}(k) = R(k-1,k) \quad \text{and} \tag{37}$$

$R_g(k) = R(k-1,k-1)$.

In the following section, we address the mathematical problems of using the optimal predictor derived in this section for obtaining the frame estimates. The equation for the optimal predictor, Equation (34), assumes that the frame to be estimated is available for deriving the autocorrelation between previous frames represented by g(k) and f(k). Hence, in practice, one must somehow estimate this autocorrelation. We do this by assuming that the image sequence is represented by a zero-mean separable Markov process which is wide-sense stationary. Specifically, the equations for a suboptimal predictor are derived as follows: The optimal predictor is given by Equation (34), i.e.,

$$B^{+}(k) = R_{gf}(k)^T R_g(k)^{-1} . \tag{34}$$

In the case of the next frame prediction using only the last frame received, i.e., $\tilde{f}(k+1)$ using only f(k), the optimal predictor is given by

$$B^{\dagger}(k) = E\{f(k)\, f(k-1)^T\}\, E\{f(k-1)\, f(k-1)^T\}^{-1} \tag{38}$$

$$= \{f(k)\, f(k-1)^T\}\, \{f(k-1)\, f(k-1)^T\}^{-1} . \tag{39}$$

We derive a suboptimal predictor by using the wide-sense stationarity on the covariance stationarity of the image sequence as follows:

Using Equations (18) and (39), the frame estimate is

$$\tilde{f}(k) = \{f(k-1)\, f(k-2)^T\}\, \{f(k-1)\, f(k-1)^T\}^{-1}\, f(k-1) . \tag{40}$$

As we see, this approach would require at least a couple of frames in order to estimate the next frame of the sequence Equation (34) assumes that an inverse of $R_g(k)$ exists. The mathematical details of the problem are discussed in the next section.

4.3. Application of Singular Value Decomposition (SVD) to the Image Processing Problem

The equation for the optimal feedback predictor, $B^{\dagger}$, given by Equation (34), assumes that an inverse of $R_g(k)$ exists. When we apply the adaptive predictor algorithm for frame prediction using the last two frames received according to Equation (40), we find that the matrix $R_g(k)$ is singular. This is because it has many zero eigenvalues as seen in Figure 5a. We then try to make it nonsingular by adding a small constant to the diagonal of $R_g(k)$. Figure 5b shows the resulting eigenvalues. It is obvous that the matrix is so ill-conditioned that it can not be inverted. In such problems involving singular matrices, one would compute the generalized inverse which is the minimum norm solution. Equation (40) may be rewritten as

$$\bar{f}(k) = [f(k-1)\, f(k-2)^T]\,(f(k-1)^T)^{\dagger}\,. \tag{41}$$

where † represents the Moore-Penrose pseudoinverse. For a matrix A, the Moore-Penrose inverse is given by

$$A^{\dagger} = (A^T A)^{-1} A^T\,. \tag{42}$$

Since the matrix $[f(k-1)f(k-1)^T]$ is singular, its rank is less than the order of f(k-1) or the number of unknowns. Hence, there is no unique inverse for this matrix. In the case of an underconstrained problem requiring a solution of a set of linear equations of the form

$$Ax = b\,,$$

we use singular value decomposition (SVD) which guarantees a minimum norm solution to the set of equations represented by $Ax=b$. Hence, to derive an optimal predictor, we use SVD of $f(k-1)^T$ to find a unique solution to an otherwise indeterminate problem.

The SVD problem involves spectral decomposition of the matrix, A, in Equation (42) as follows. In the case of real matrix A,

$$A = U \Sigma V^T\,, \tag{43}$$

where U and V are unitary matrices such that the columns of U and V are composed of a set of orthonormal eigenvectors. U is the row eigenvector system of A and V is the column eigenvector system of A. The matrix Σ is a diagonal matrix which has singular values of A (the positive square-roots of the non-zero eigenvalues λ_i of $A^T A$) on its diagonal. The eigenvectors u_i are the spectral components of the observation space and the eigenvectors v_i are the spectral components of

```
E.VAL.   1 = 0.000E+00        E.VAL.  17 = 0.272E+01
E.VAL.   2 = 0.000E+00        E.VAL.  18 = 0.478E+01
E.VAL.   3 = 0.000E+00        E.VAL.  19 = 0.755E+01
E.VAL.   4 = 0.000E+00        E.VAL.  20 = 0.120E+02
E.VAL.   5 = 0.000E+00        E.VAL.  21 = 0.178E+02
E.VAL.   6 = 0.000E+00        E.VAL.  22 = 0.205E+02
E.VAL.   7 = 0.000E+00        E.VAL.  23 = 0.287E+02
E.VAL.   8 = 0.000E+00        E.VAL.  24 = 0.317E+02
E.VAL.   9 = 0.000E+00        E.VAL.  25 = 0.328E+02
E.VAL.  10 = 0.000E+00        E.VAL.  26 = 0.804E+02
E.VAL.  11 = 0.000E+00        E.VAL.  27 = 0.930E+02
E.VAL.  12 = 0.000E+00        E.VAL.  28 = 0.119E+03
E.VAL.  13 = 0.000E+00        E.VAL.  29 = 0.159E+03
E.VAL.  14 = 0.000E+00        E.VAL.  30 = 0.247E+03
E.VAL.  15 = 0.000E+00        E.VAL.  31 = 0.722E+04
E.VAL.  16 = 0.000E+00        E.VAL.  32 = 0.522E+07
```

(a)

```
E.VAL.   1 = 0.000E+00        E.VAL.  17 = 0.360E+04
E.VAL.   2 = 0.000E+00        E.VAL.  18 = 0.360E+04
E.VAL.   3 = 0.000E+00        E.VAL.  19 = 0.361E+04
E.VAL.   4 = 0.000E+00        E.VAL.  20 = 0.361E+04
E.VAL.   5 = 0.000E+00        E.VAL.  21 = 0.362E+04
E.VAL.   6 = 0.000E+00        E.VAL.  22 = 0.362E+04
E.VAL.   7 = 0.000E+00        E.VAL.  23 = 0.363E+04
E.VAL.   8 = 0.000E+00        E.VAL.  24 = 0.363E+04
E.VAL.   9 = 0.000E+00        E.VAL.  25 = 0.363E+04
E.VAL.  10 = 0.000E+00        E.VAL.  26 = 0.368E+04
E.VAL.  11 = 0.000E+00        E.VAL.  27 = 0.369E+04
E.VAL.  12 = 0.000E+00        E.VAL.  28 = 0.372E+04
E.VAL.  13 = 0.000E+00        E.VAL.  29 = 0.376E+04
E.VAL.  14 = 0.000E+00        E.VAL.  30 = 0.385E+04
E.VAL.  15 = 0.000E+00        E.VAL.  31 = 0.108E+05
E.VAL.  16 = 0.000E+00        E.VAL.  32 = 0.522E+07
```

(b)

Figure 5. Singularity of the image processing matrices.
(a) The eigenvalues of $[f(k-1)\,f(k-1)^T]$, where f(k-1) corresponds to the first 32×32 block image of Image 2.
(b) The eigenvalues of $[f(k-1)\,f(k-1)^T + 3600 * I]$, where f(k-1) corresponds to the first 32×32 block of Image 2 and I is the identity matrix.

the parameter space. The value of each λ_i determines the amplitude of the corresponding spectral components u_i and v_i. The generalized inverse to the matrix A of Equation (43) is defined as

$$A = V\,\Sigma^T\,U^T\,. \tag{44}$$

Using Equation (43), the generalized inverse solution to the set of linear equations, $Ax = b$, can be expressed as a linear combination of the eigenvectors v as

$$x = \sum_{i=1}^{k} v_i \left[\frac{1}{\lambda_i}\,u_i^T\,b\right]. \tag{45}$$

The weighting factors in this linear combination are the quantities in the square brackets. The products $u_i^T b$ represent the amplitude of the ith spectral component contained in the observations, b. This quantity is divided by the response λ_i. Thus, the quantity in the brackets represents the amount of the ith spectral component contributing to the solution parameter-vector x. The equation shows that the observational errors or model errors in defining the system dynamics are amplified by a factor, $1/\lambda_i$. Thus eigenvectors that are poorly represented in the sense that they are associated with small values of λ_i cannot be determined as reliably as the better represented eigenvectors. For limiting the influence of the eigenvectors with very small eigenvalues, one approach is to eliminate the small eigenvalues by simply adjusting the value of the apparent rank of the matrix A. This is done via SVD. We now apply this approach to try to find an approximate inverse of $f(k-1)^T$ which would give an acceptable prediction.

When we use the SVD of $f(k-1)^T$, we find that an acceptable solution to the problem is still not possible. This is because of the ill-conditioned nature of the matrix which makes its non-zero singular values vary over a wide range as is seen from Figure 6. The values in Figures 5-7 relate to prediction of Image 3 from Image 2 and Image 1. When we attempt to restore the image by discarding the zero singular values and use the non-zero ones, we find that the smaller singular values effectively increase the contribution of the noise term and make the visual quality of the image unacceptable. The use of only the highest singular value provides acceptable values for the objective criteria but results in minimizing a lot of detail in the reconstructed picture. We attempt

```
S.V.  1 =  2285.074        S.V. 17 =       0.000
S.V.  2 =    26.524        S.V. 18 =       0.000
S.V.  3 =    10.268        S.V. 19 =       0.000
S.V.  4 =     6.970        S.V. 20 =       0.000
S.V.  5 =     5.078        S.V. 21 =       0.000
S.V.  6 =     3.395        S.V. 22 =       0.000
S.V.  7 =     2.349        S.V. 23 =       0.000
S.V.  8 =     1.788        S.V. 24 =       0.000
S.V.  9 =     0.000        S.V. 25 =       0.000
S.V. 10 =     0.000        S.V. 26 =       0.000
S.V. 11 =     0.000        S.V. 27 =       0.000
S.V. 12 =     0.000        S.V. 28 =       0.000
S.V. 13 =     0.000        S.V. 29 =       0.000
S.V. 14 =     0.000        S.V. 30 =       0.000
S.V. 15 =     0.000        S.V. 31 =       0.000
S.V. 16 =     0.000        S.V. 32 =       0.000
```

(a)

```
106.000 102.000 101.000  98.000
95.000  98.000  93.000  94.000
109.000 104.000 103.000 100.000
96.000  98.000  94.000  95.000
104.000 106.000 105.000 100.000
98.000  97.000  95.000  96.000
104.000 109.000 104.000 101.000
98.000 100.000  96.000  99.000
99.000 109.000 103.000 103.000
97.000 101.000  96.000  99.000
101.000 107.000 103.000 103.000
97.000 101.000  96.000  97.000
100.000 107.000 103.000 103.000
97.000 100.000  96.000  99.000
101.000 107.000 103.000 103.000
98.000  98.000  97.000 101.000
```

(b)

Figure 6. Singularity of the image processing matrices.
(a) The singular values of $f(k-1)^T$, where f(k-1) corresponds to the first 32×32 block of Image 2. (b) Pixel intensities corresponding to the first 8×8 block of Image 2.

S.V.	1 =	0.171 E +24		S.V.	17 =	0.000 E +00	
S.V.	2 =	0.108 E +24		S.V.	18 =	0.000 E +00	
S.V.	3 =	0.822 E +23		S.V.	19 =	0.000 E +00	
S.V.	4 =	0.680 E +23		S.V.	20 =	0.000 E +00	
S.V.	5 =	0.388 E +23		S.V.	21 =	0.000 E +00	
S.V.	6 =	0.656 E +16		S.V.	22 =	0.000 E +00	
S.V.	7 =	0.253 E +16		S.V.	23 =	0.000 E +00	
S.V.	8 =	0.160 E +16		S.V.	24 =	0.000 E +00	
S.V.	9 =	0.159 E +10		S.V.	25 =	0.000 E +00	
S.V.	10 =	0.182 E +09		S.V.	26 =	0.000 E +00	
S.V.	11 =	0.572 E +08		S.V.	27 =	0.000 E +00	
S.V.	12 =	0.253 E +08		S.V.	28 =	0.000 E +00	
S.V.	13 =	0.111 E +02		S.V.	29 =	0.000 E +00	
S.V.	14 =	0.361 E +01		S.V.	30 =	0.000 E +00	
S.V.	15 =	0.184 E +01		S.V.	31 =	0.000 E +00	
S.V.	16 =	0.518 E +00		S.V.	32 =	0.000 E +00	

(a)

0.333	-1.250	0.000	-0.500
-1.750	2.000	-1.750	0.250
3.250	-0.800	0.400	0.600
-1.400	1.400	-1.000	0.600
-1.750	0.400	1.400	-0.800
0.200	-0.600	-0.600	0.000
0.000	2.600	-0.400	-0.200
-0.800	1.600	-1.200	2.200
-4.250	3.600	-1.400	1.600
-2.200	2.000	-1.600	2.200
-0.750	1.600	-0.800	1.200
-2.000	2.000	-1.200	0.200
-2.250	2.200	-0.800	1.200
-2.000	1.600	-1.600	1.600
-0.750	2.000	-0.600	1.200
-1.000	-0.200	-0.800	2.200

(b)

Figure 7. Singularity of the image processing matrices.
(a) The singular values of $f(k-1)^T$, where f(k-1) corresponds to the first 32×32 block of Image 2 after subtracting the local mean taken over a 5×5 pixel window. (b) Pixel intensities corresponding to the first 8×8 block of Image 2 after subtracting the local mean.

to lower the range of the non-zero singular values by adjusting the matrix f(k-1) as follows. Instead of choosing f(k-1) to represent the pixel intensities, we subtract from each pixel value its local mean taken over a 5×5 pixel window centered over that pixel. The resulting values for an 8×8 pixel subblock and the corresponding singular values of $f(k-1)^T$ are shown in Figure 7. The new matrix is still seen to be ill-conditioned, preventing an acceptable solution to the inverse problem.

To eliminate the mathematical complexities involved in the derivation of the optimal predictor, we derive a suboptimal predictor which still uses the past frames to form a prediction. This is not unreasonabe, since any on-line estimation algorithm must be computationally simple to implement. The equation for the suboptimal predictor for the next frame prediction using the last two frames is as follows:

$$\tilde{f}(k) = [f(k-1)\, f(k-2)^T]\, f(k-1) / \| f(k-1) \|^2 \qquad (46)$$

where $\| \cdot \|$ represents the Frobenius norm .

For a matrix A,

$$\| A \| = \text{trace}\,(A^T A)\,.$$

The results derived via the suboptimal predictor are discussed briefly in the following section and in greater detail in Chapter 6.

4.4. Performance Evaluation

The performance of the suboptimal predictor is summarized in Table 8 for one-frame ahead prediction, using the last two frames received. The corresponding results for the fixed predictor are given in Table 4, and for frame refresh in Table 7. The estimates derived via this pedictor are shown in Chapter 6. From an evaluation of the mean-square error and SNR values, we can see that the suboptimal predictor which uses the past statistics to derive the estimate, matches the performance of the fixed predictor. The fixed predictor is seen to give better criterion values than

DESCRIPTION	BLOCK #	[illegible]	[illegible]	[illegible]	[illegible]	[illegible] AVG SNR dB
IMAGE 3 FROM IMAGE 2 & IMAGE 1	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	33.2
	2	[illegible]	[illegible]	[illegible]	[illegible]	
	3	[illegible]	[illegible]	[illegible]	31.2	
	4	2.67	14.3	0.14	36.6	
IMAGE 4 FROM IMAGE 3 & IMAGE 2	1	5.16	47.9	0.41	31.3	31.7
	2	2.96	22.3	0.21	34.7	
	3	8.15	94.1	0.77	28.4	
	4	3.03	17.5	0.17	35.7	
IMAGE 5 FROM IMAGE 4 & IMAGE 3	1	6.04	57.7	0.49	30.5	32.1
	2	3.28	25.8	0.24	34.0	
	3	6.20	60.4	0.49	30.3	
	4	2.83	16.8	0.16	35.9	
IMAGE 6 FROM IMAGE 5 & IMAGE 4	1	8.18	140.0	1.22	26.7	27.0
	2	10.2	242.0	1.92	24.3	
	3	5.95	57.7	0.48	30.5	
	4	5.57	63.3	0.55	30.1	
IMAGE 7 FROM IMAGE 6 & IMAGE 5	1	9.11	147.0	1.26	26.5	25.9
	2	12.2	365.0	2.79	22.5	
	3	6.59	67.8	0.55	29.8	
	4	5.56	62.7	0.54	30.2	
IMAGE 8 FROM IMAGE 7 & IMAGE 6	1	7.38	101.0	0.83	28.1	27.3
	2	10.3	253.0	1.83	24.1	
	3	7.35	88.3	0.67	28.7	
	4	3.81	32.1	0.27	33.1	

the frame refresh in almost all cases. However, subjective quality is seen to be less acceptable than the fixed predictor. There does not appear to be a significant advantage of using this technique over frame refresh which does not require any computations. For the adaptive predictor case, if we can find a computational solution to the problem of inverting an extremely ill-conditioned matrix, then the results are expected to be better than what the fixed predictor provides. In situations where a direct pseudoinverse algorithm produces unusable restored images due to the ill-conditioned natue of the problem, one approach suggested in [12] is to apply the SVD technique to both row and column matrices separately. The use of this approach for reconstructing the images merits further investigation and is a topic for future research in this area.

CHAPTER 5

APPLICATION OF MOTION ESTIMATION FOR IMPROVING THE ACCURACY OF FRAME PREDICTION

5.1. Introduction

A large number of applications requiring image processing involve images of moving objects. For example, in our application of satellite image processing for rendezvous and docking, the scenario is characterized by both a moving target as well as a moving camera. In such cases, the nonstationarity of the images renders the image prediction a more complicated process. To accurately predict the image data in advance, one must estimate and account for the interframe motion. A number of researchers have addressed this issue of determining the motion of an object from a sequence of images. Motion compensation can be thought of as a filtering process where the interframe motion is considered a noise. To the extent that the signal and noise have different power spectra, one can filter the noise representing the motion. For a stationary random process, this could be done via Wiener filtering. However, scene dynamics cannot accurately be modeled as a stationary process. Hence, one must look for alternative means of motion compensation in the case of video images.

Dubois and Sabri [13] apply temporal filtering to detect that part of the image which is nonstationary between frames, and use motion compensation to reduce the noise in image sequences. Bowling and Jones [14] use a two-step displacement procedure to determine pixel displacements between frames. Broida and Chellappa [15] give a recursive estimation procedure for determining object motion parameters from a sequence of images. In our work, we applied the algorithm presented by Jain and Jain in [16] for measuring the displacement between consecutive frames in integer number of pixels. There are algorithms available to measure the pixel displacement with subpixel accuracy. Limb and Murphy [17] present such an algorithm for measuring the speed of moving objects from TV signals; Cafforio and Rocca suggest an algorithm for measuring small displacements in television images in [18]. The algorithm proposed by Jain and Jain entails dividing the image frame into subblocks and applying a 2-dimensional directed search procedure to find the displacement which optimizes a certain criterion. In our application, the optimization criterion is chosen as the mean- square error between the corresponding blocks of the consecutive frames. This is consistent with the criteria used for a comparison of the fixed and adaptive predictor results. This displacement is applied to the frame estimate to improve the accuracy of prediction. The details of the algorithm are summarized in the following section.

5.2. Motion Estimation

Jain and Jain [16] give an algorithm for measuring the interframe motion for digitized images. This procedure makes use of the fact that in practice, a large part of the motion in a scene can be approximated by piecewise translation of several areas of a frame. The procedure consists of segmenting an image into fixed size, rectangular blocks. It is assumed that each of these blocks is undergoing independent translation. Then the rotation and zooming, etc., of larger objects in the scene are approximated by a piecewise translation of these smaller blocks.

We used 32×32 subblocks consistent with the block sizes used for other algorithm computations. Let U be an N×N size block of a given frame in a sequence. Let V be an (N+2p)×(N+2p) size subblock of the consecutive frame of the sequence, centered at the same spatial location as U. Here, p is the maximum displacement allowed in either direction in integer number of pixels. A mean distortion function, the mean-square error between U and V, is used as the criterion for determining the direction of minimum distortion (DMD) between the two frames; in other words, the displacement D that minimizes the mean-square error between U and V.

$$D(i,j) = \frac{1}{N \times N} \sum_{m=1}^{N} \sum_{n=1}^{N} g(u(m,n) - v(m+i,n+j)) , \quad -p \leqslant i,j \leqslant p , \quad \text{where}$$

$g(x) = x^2$ corresponds to the mean-square error between U and V. In general, g(x) can be any positive and increasing function of x. We use the mean-square error criterion since it is consistent with the performance measures used in this work. Also, this allows us to obtain results about improvement in frame prediction in mean-square error (or signal-to-noise ratio) terms.

The following assumption is made about the mean distortion function. If $D(q,l) = \min_{i,j}\{D(i,j)\}$ then for $m = i-q$, $n = j-l$, the functions

$$D_1(|m|,|n|) = D(i,j) - D_0(q,l), \quad m \geqslant 0, \; n \geqslant 0$$

$$D_2(|m|,|n|) = D(i,j) - D_0(q,l), \quad m \geqslant 0, \; n \leqslant 0$$

$$D_3(|m|,|n|) = D(i,j) - D_0(q,l), \quad m \leqslant 0, \; n \leqslant 0$$

$$D_4(|m|,|n|) = D(i,j) - D_0(q,l), \quad m \leqslant 0, \; n \geqslant 0,$$

where $|\cdot|$ represents the absolute value. The above assumption means that the distortion function increases monotonically as we move away from the direction of the minimum distortion.

With the above assumption, the algorithm [16] uses a 2-dimensional directed search method for finding the DMD. The search consists of testing five locations in a frame at a time and successively reducing the area of search until the plane of search reduces to a 3×3 block. In the final step, all *nine locations are searched to find the* DMD *and the minimum* mean-square error. The DMD and the minimum mean-square error for the entire frame are obtained by averaging over all subblocks.

The algorithm is as follows:

For any integer $p>0$,

$$N(p) = \{(i,j) : -p \leqslant i,j \leqslant p)$$

$$M(p) = \{(0,0), (p,0), (0,p), (-p,0), (0,-p)\}.$$

Step 1: (Initialization)

$$D(i,j) = \infty, \quad (i,j) \notin N(p)$$

$n' = [\log_2 p]$, [ ] is a lower integer truncation function

$$n = \max\{2, 2^{n'-1}\}$$

$$q = l = 0.$$

Step 2: $M'(n) \leftarrow M(n)$.

Step 3: Find $(i,j) \in M'(n)$ such that $D(i+q,j+l)$ is minimum. If $i=0$ and $j=0$, go to Step 5; otherwise, go to Step 4.

Step 4: $q \leftarrow q+i$, $l \leftarrow l+j$; $M(n) \leftarrow M'(n) - (-i,-j)$; go to Step 3.

Step 5: $n \leftarrow n/2$. If $n=1$, go to Step 6; otherwise, go to Step 2.

Step 6: Find $(i,j) \in N(1)$ such that $D(i+q,j+l)$ is minimum. The DMD is then given by $q \leftarrow q+i$, $l \leftarrow l+j$.

The proof of the algorithm can be found in [16].

5.3. Performance Evaluation

The above algorithm is used to derive the displacement between the estimates derived by the fixed predictor algorithm and the actual frames. The results for some of these estimates are tabulated in Table 9. The improvement in the prediction is apparent from the reduction in the MSE and the corresponding increase in the SNR. It is shown that an improvement of approximately 2 dB is obtained when we compensate for the motion.

We find that even though the displacement measured matches well with the actual displacement on a subblock basis, the approximation involved in measuring the displacement in integer number of pixels degrades the accuracy of the overall displacement when averaged over the entire image. The displacement algorithm can be applied to the entire image without too much computational burden. However, we apply it to 32×32 blocks for consistency with the other results. It is obvious that where averaging is expected over many subblocks, we require an algorithm that gives the displacement with fractional pixel accuracy. The works of Limb and Murphy [17], and Cafforio and Rocca [18] are concerned with the problem of measuring small displacements in television images.

In practice, for the purpose of on-line motion compensation, we would estimate the interframe displacement between the two most recently received frames and apply it to the

Table 9. Effect of Motion Compensation on Interframe Prediction
One-Step Ahead Prediction

DESCRIP-TION	BLK #	WITHOUT MO-TION COMPEN-SATION		WITH MOTION COMPENSATION		IMPROVEMENT IN SNR	OVERALL DIS-PLACEMENT
		MSE	SNR	MSE	SNR		
IMAGE 2	1	18.6	35.4	6.7	39.9	2.5	(-2,-4)
FROM	2	8.8	38.7	6.8	39.8		
	3	14.2	36.6	8.5	38.8		
IMAGE 1	4	5.5	40.8	4.7	35.3		
IMAGE 3	1	40.0	32.1	19.2	35.3	2.1	(3,-1)
FROM	2	15.1	36.3	9.6	38.3		
	3	48.0	31.3	34.7	32.7		
IMAGE 2	4	11.3	37.6	7.23	39.6		
IMAGE 4	1	54.0	30.8	29.1	33.5	2.3	(0,-2)
FROM	2	14.6	36.5	8.7	38.8		
	3	42.4	31.9	27.9	33.7		
IMAGE 3	4	9.6	38.3	6.5	40.0		
IMAGE 5	1	57.7	30.5	21.1	34.9	3.0	(-1,4)
FROM	2	16.7	35.9	83.7	28.9		
	3	37.5	32.4	24.5	34.2		
IMAGE 4	4	10.0	38.1	6.5	40.0		
IMAGE 6	1	166.0	25.9	84.9	28.8	3.2	(1,0)
FROM	2	227.0	24.6	94.3	28.4		
	3	55.8	30.7	34.2	32.8		
IMAGE 5	4	54.2	30.8	25.0	34.2		
IMAGE 7	1	71.9	29.6	36.8	32.5	3.4	(1,-2)
FROM	2	121.0	27.3	46.5	31.5		
	3	46.0	31.5	22.9	34.6		
IMAGE 6	4	20.3	35.2	10.3	38.1		
IMAGE 8	1	87.4	28.7	43.3	31.8	3.1	(-1,-1)
FROM	2	193.0	25.3	81.7	29.0		
	3	69.3	29.7	44.5	31.7		
IMAGE 7	4	25.2	34.1	13.3	36.9		

Table 9. (Continued) Effect of Motion Compensation on Interframe Prediction
Two- and Three-Step Ahead Prediction

DESCRIPTION	BLK #	WITHOUT MOTION COMPENSATION		WITH MOTION COMPENSATION		IMPROVEMENT IN SNR	OVERALL DISPLACEMENT
		MSE	SNR	MSE	SNR		
IMAGE 3 FROM IMAGE 1	1	41.2	32.0	18.4	35.5	2.6	(2, 1)
	2	15.0	36.4	6.8	39.8		
	3	57.0	30.6	33.5	32.9		
	4	13.3	36.9	8.2	38.2		
IMAGE 4 FROM IMAGE 1	1	46.7	31.4	20.9	35.0	2.7	(1,-3)
	2	14.8	36.4	10.8	37.8		
	3	89.4	28.6	50.4	31.1		
	4	15.7	36.2	10.0	38.2		

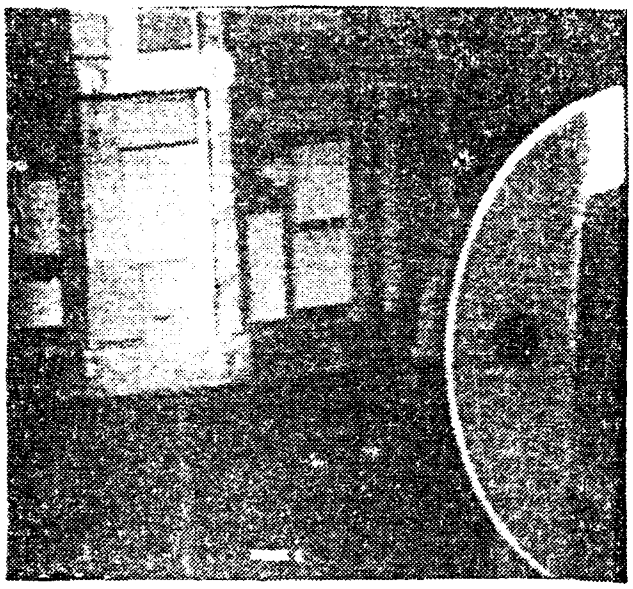
reconstructed frame. A better result is expected when we use displaced frames as opposed to the actual frames for deriving both fixed and adaptive predictors. Since the displacement algorithm can be applied to the entire image, an on-line motion compensation for an adaptive prediction appears to be feasible and computationally simple. These results are discussed further in Chapter 6.

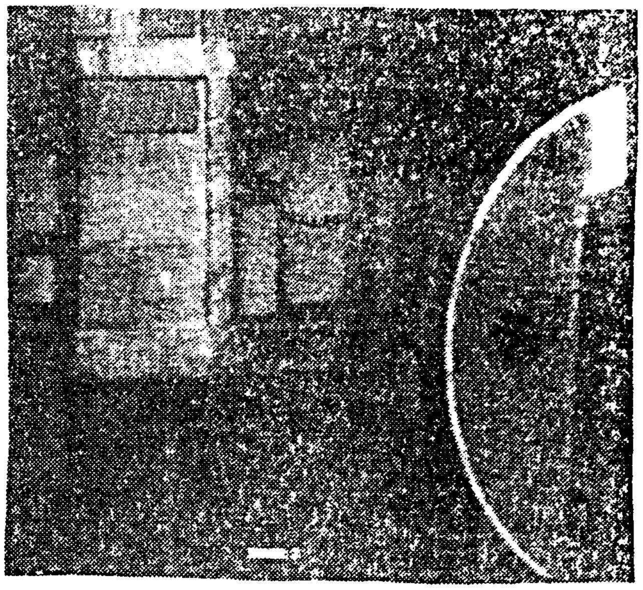
CHAPTER 6

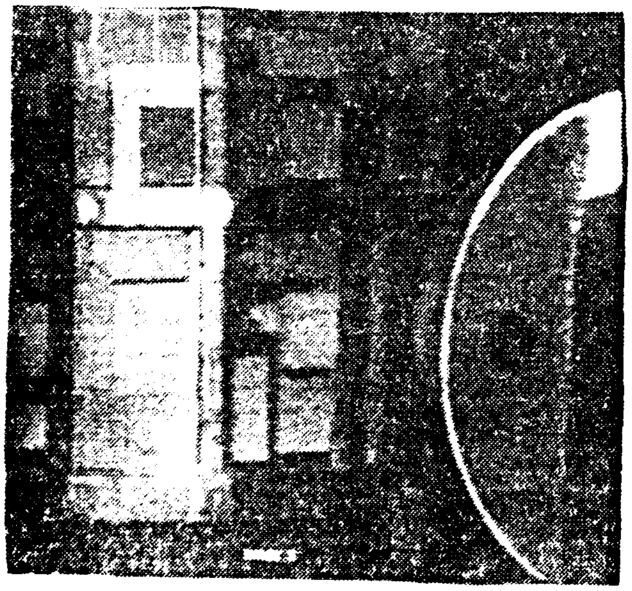
DISCUSSION OF THE SIMULATION RESULTS

In this chapter, we discuss the results obtained from the fixed and the adaptive predictor algorithms. As described in Chapter 1, in order to demonstrate the performance of the algorithms, we use a set of 8 frames from a video tape of a spacecraft-to-spacecraft docking simulation. Figure 8 shows the 8 frames. These frames are not successive frames of the video tape with the typical frame rate being 30 frames per second, but were taken at different stages in the docking. We selected frames representing far-, mid-, and close-range in order to determine the robustness of the approaches for different situations. For example, a change in the camera azimuth or elevation introduces new and, therefore, unpredictable area due to the presence of unmodeled dynamics. The first 3 frames represent the far-range. The zooming effect is present between images 5 and 6. In practice, however, a remote pilot can expect successive frames with small interframe displacement due to the very small velocity of the spacecraft in the docking phase. Hence, the results obtained in this work represent conservative estimates.

The results of the fixed predictor for up to 3 steps of prediction, and the next frame prediction using one, two, and three of the most recent frames are summarized in Tables 1-5 using the criteria defined in Chapter 2, namely, MABSE, MSE, % NMSE, and SNR. The corresponding pictures of the reconstructed images are shown in Figures 8-13, respectively. In some cases, we present only the first picture of the sequence to limit the total number of pictures, but the analysis was carried out for all cases in all categories and the results are summarized below the figures to give an idea of the relative subjective merit of the other estimates. For comparison, the results of the frame refresh approach of prediction are summarized in Tables 6 and 7 for one and two steps of prediction, respectively. The side effects of processing the images in subblocks cause minor degradation in the reconstructed picture quality, mainly in the form of a border effect (more noticeable in the disc towards the right). This is caused by the partitioning of images into 32×32 subblocks for processing and thereafter pasting the processed blocks to obtain the complete image. On the


(a)


(b)


(c)

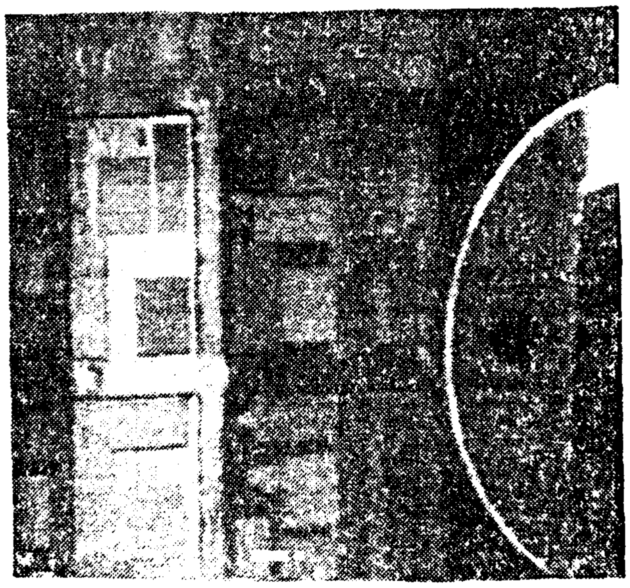
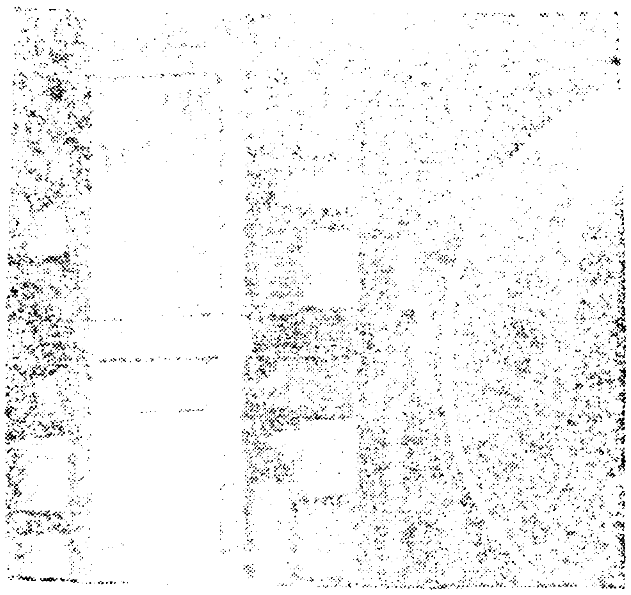
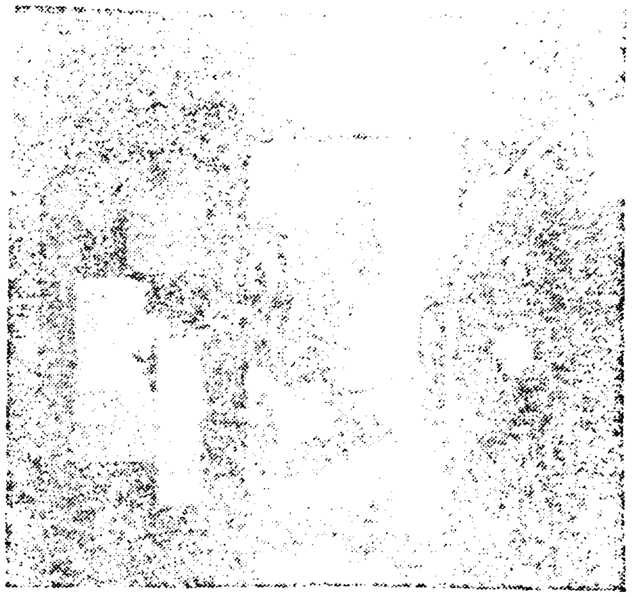

(d)

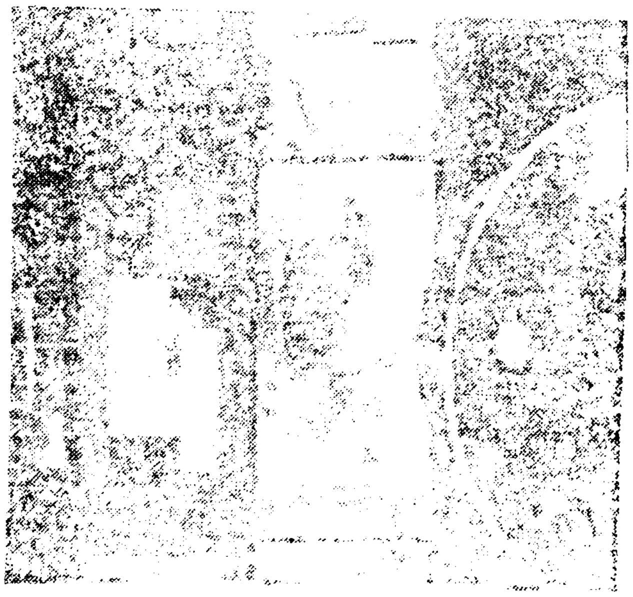
Figure 8. Image sequence representing spacecraft docking scenario. (a)-(d) represent images 1-4, respectively.



(a)



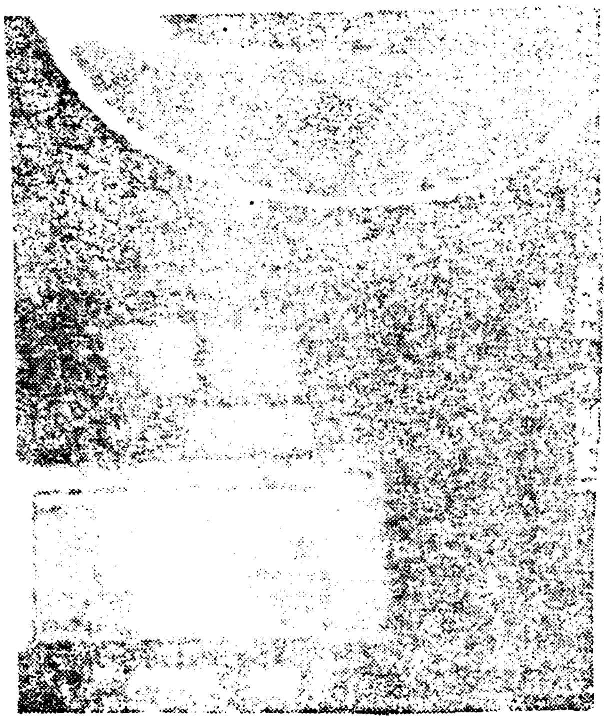
[illegible]

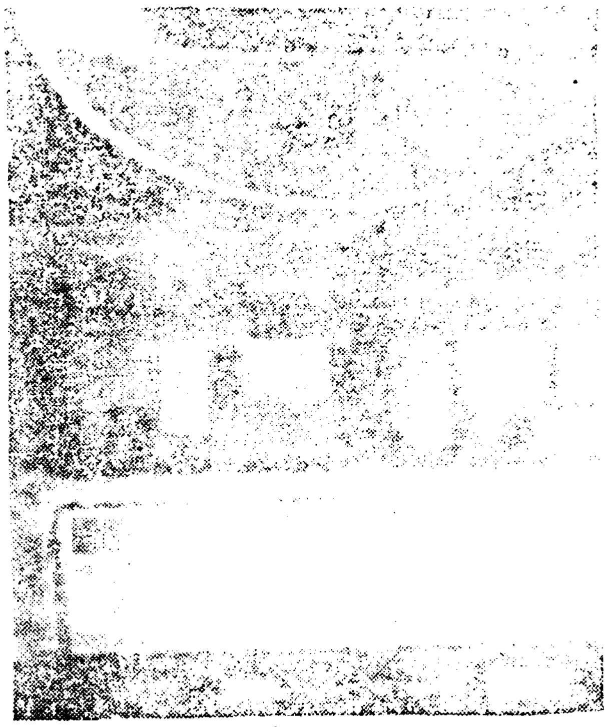


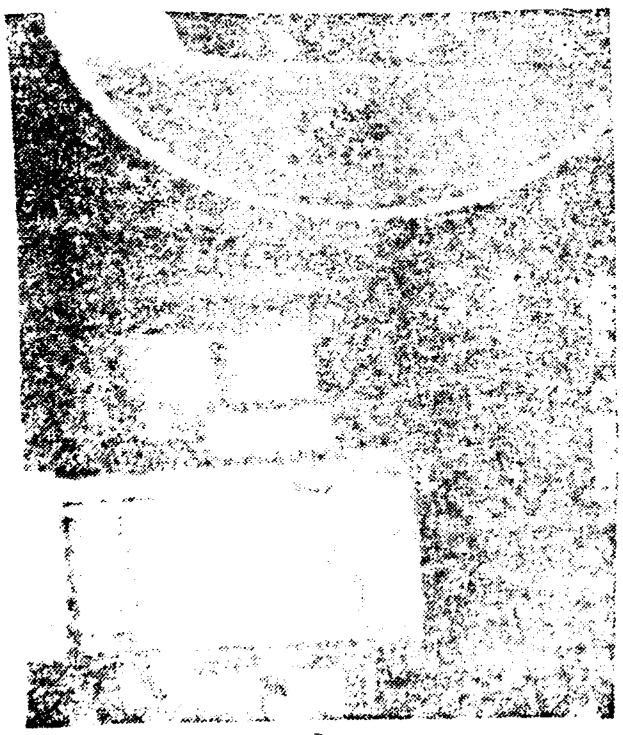
[illegible]

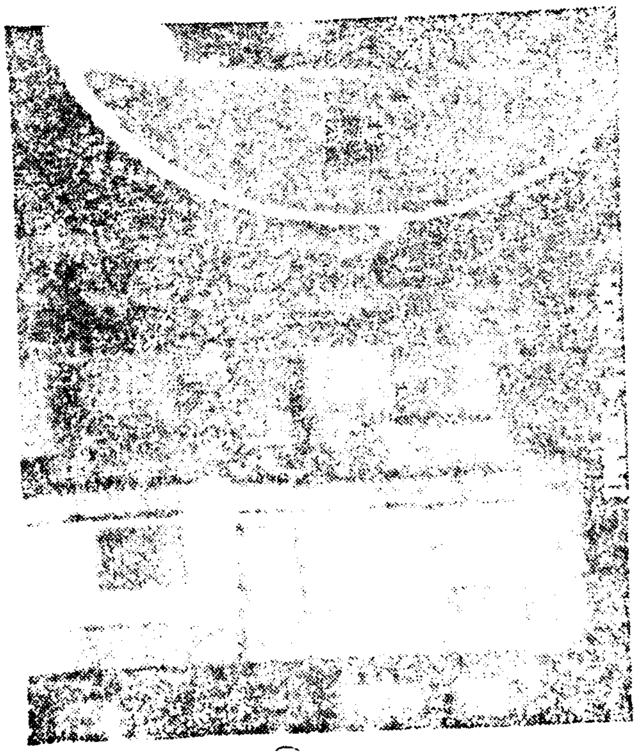
Figure 5 (Continued) Images [illegible] pre[illegible] [illegible] [illegible]
e) [illegible] experiment [illegible] [illegible]

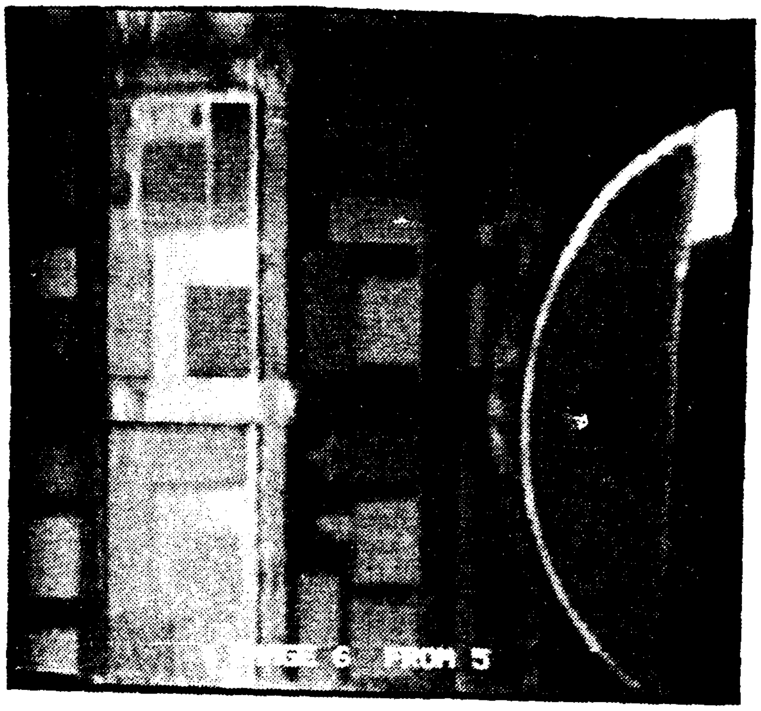
Figure 9. Next-frame prediction via fixed predictor technique. The resulting images and the error between the original and the reconstructed frame represented by the signal-to-noise ratio (SNR). (a) Image 2 from Image 1, SNR = 37.4 dB. (b) Image 3 from Image 2, SNR = 33.6 dB. (c) Image 4 from Image 3, SNR = 33.3 dB. (d) Image 5 from Image 4, SNR = 33.3 dB.


(b)

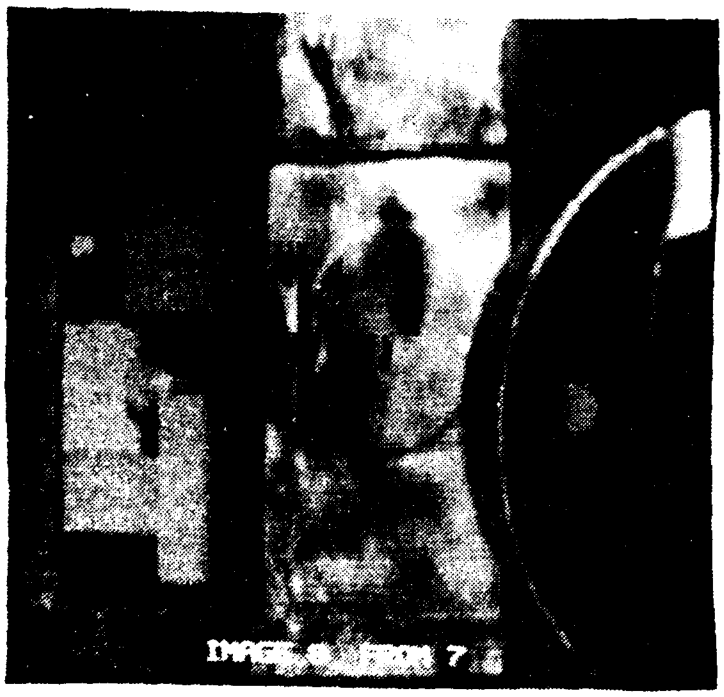

(d)


(a)


(c)



(e)



(f)

Figure 9. (continued) Next-frame prediction via fixed predictor technique. The resulting images and the error between the original and the reconstructed frame represented by the signal-to-noise ratio (SNR). (e) Image 6 from Image 5, SNR = 27.1 dB. (f) Image 8 from Image 7, SNR = 28.4 dB.

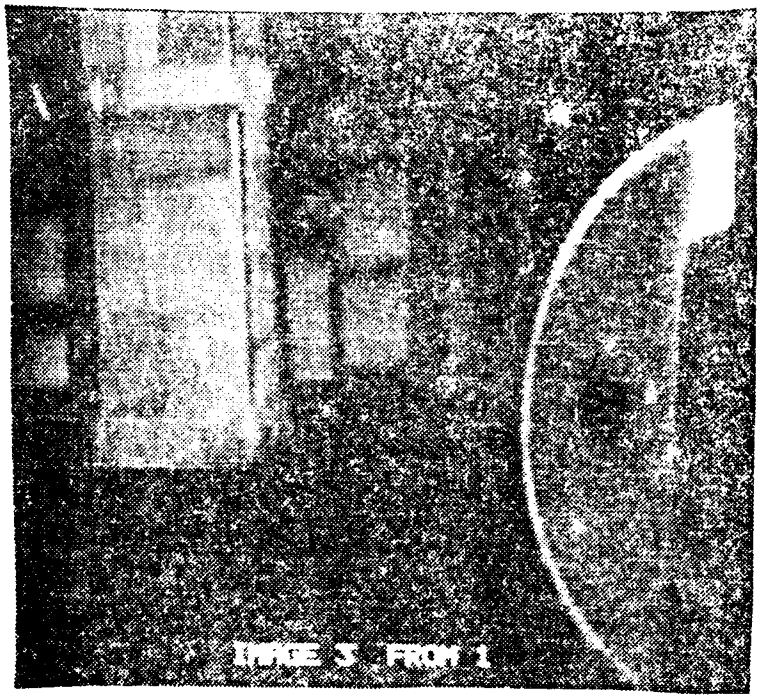


Figure 10. Two-frame ahead prediction via fixed predictor technique. The resulting image and the error between the original and the reconstructed frame represented by the signal-to-noise ratio (SNR). Image 3 from Image 1, SNR = 33.1 dB.

Note: SNR values for the other images are as follows: Image 4 from 2, SNR = 31.7 dB; Image 5 from 3, SNR = 32.3 dB; Image 6 from 4, SNR 27.1 dB; Image 7 from 6, SNR = 26.1 dB; Image 8 from 6, SNR = 27.2 dB.

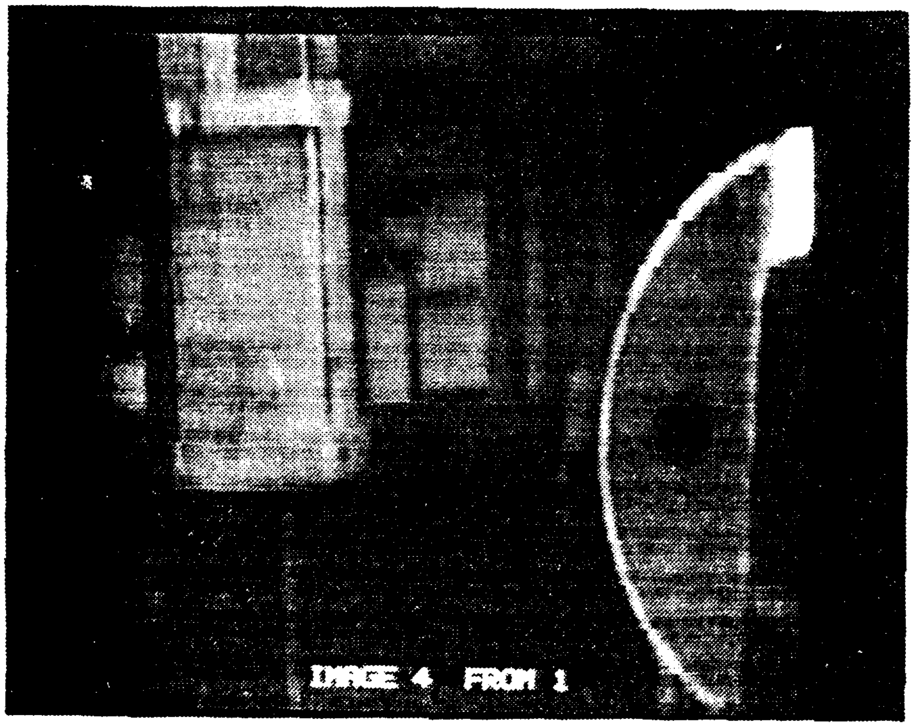
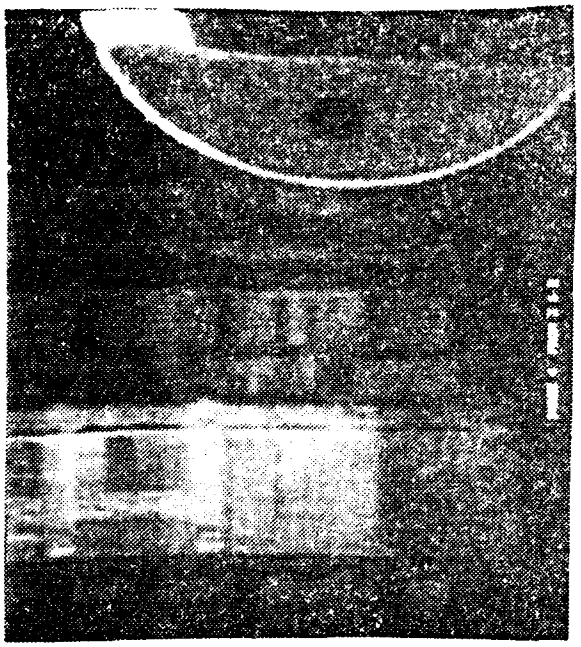
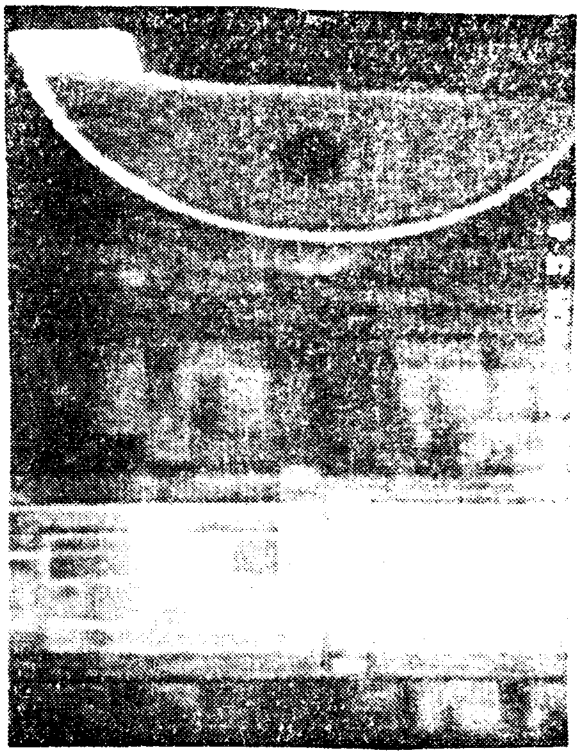


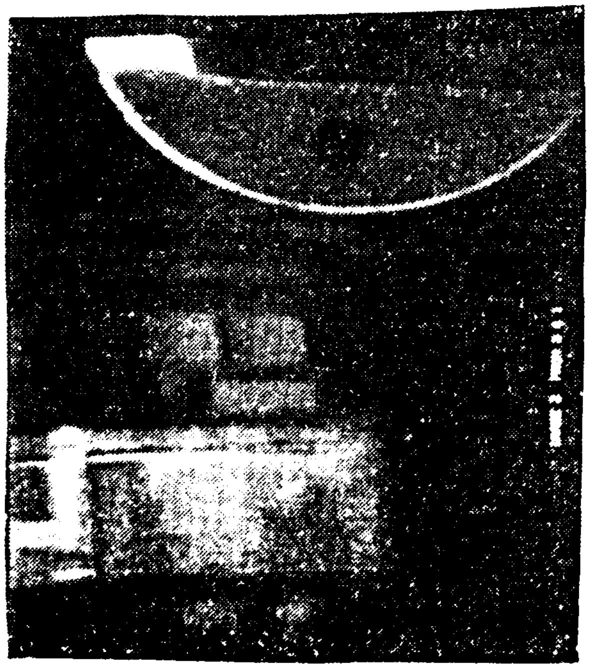
Figure 11. Three-frame ahead prediction via fixed predictor technique. The resulting image and the error between the original and the reconstructed frame represented by the signal-to-noise ratio (SNR). Image 4 from Image 1, SNR = 31.9 dB.

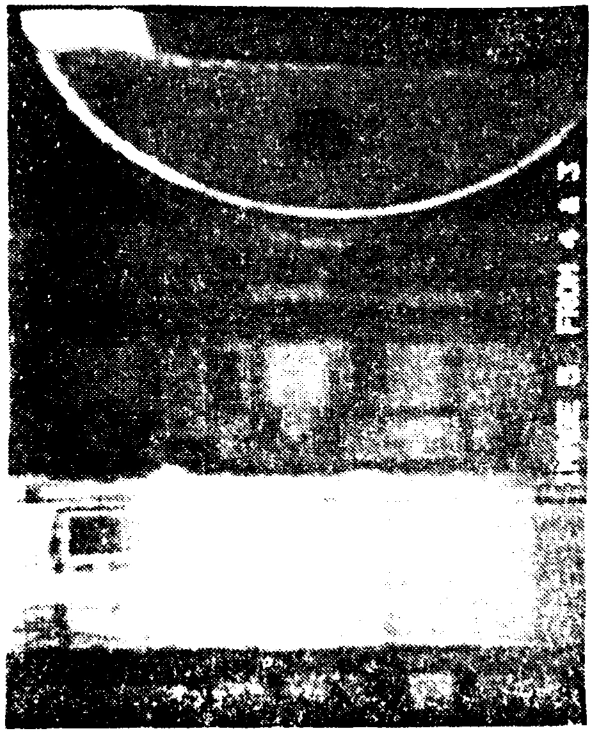
Note: SNR values for the other images are as follows: Image 5 from Image 2, SNR = 31.3 dB; Image 6 from Image 3, SNR = 27.1 dB; Image 7 from Image 4, SNR = 26.5 dB; Image 8 from Image 5, SNR = 24.9 dB.

Figure 12. Next-frame prediction via fixed predictor technique using last two frames received. The resulting images and the error between the original and the reconstructed frame represented by the signal-to-noise ratio (SNR).
(a) Image 3 from Images 2 and 1, SNR = 33.7 dB. (b) Image 4 from Images 3 and 2, SNR = 33.4 dB. (c) Image 5 from Images 4 and 3, SNR = 33.7 dB. (d) Image 6 from Images 5 and 4, SNR = 27.4 dB.


(b)


(d)


(a)

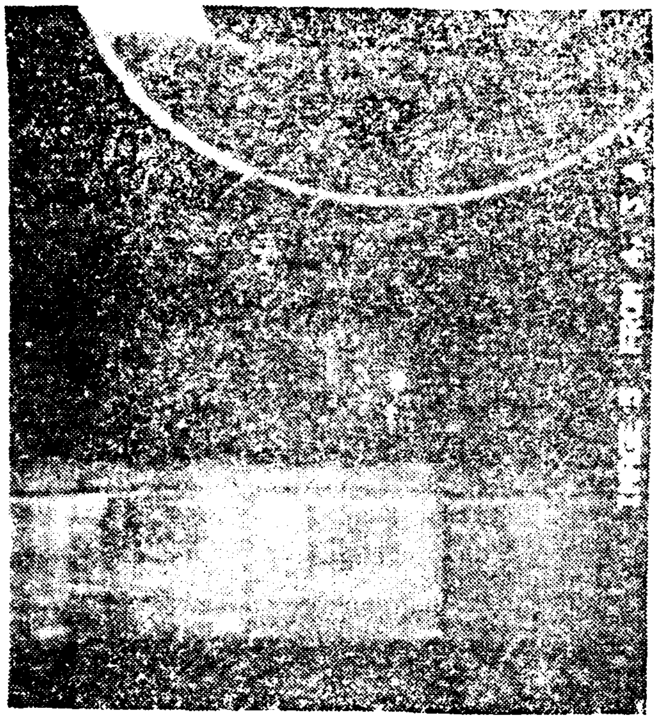

(c)

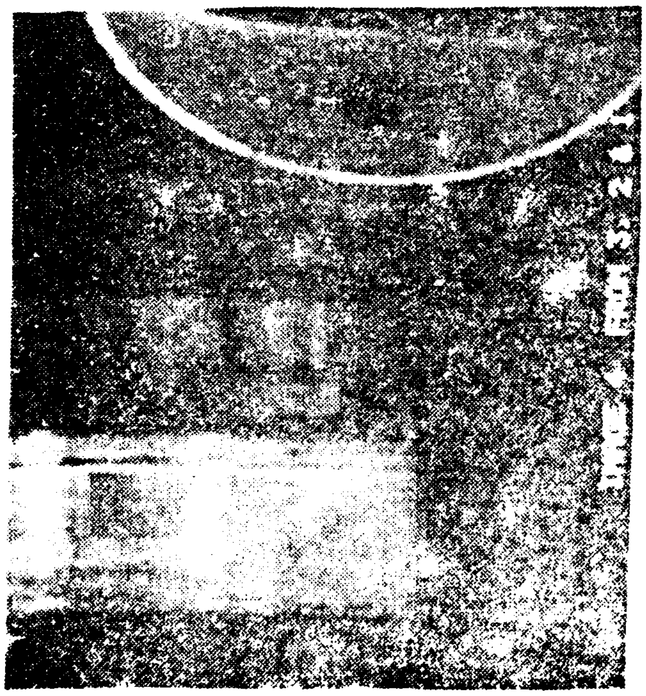
(e)

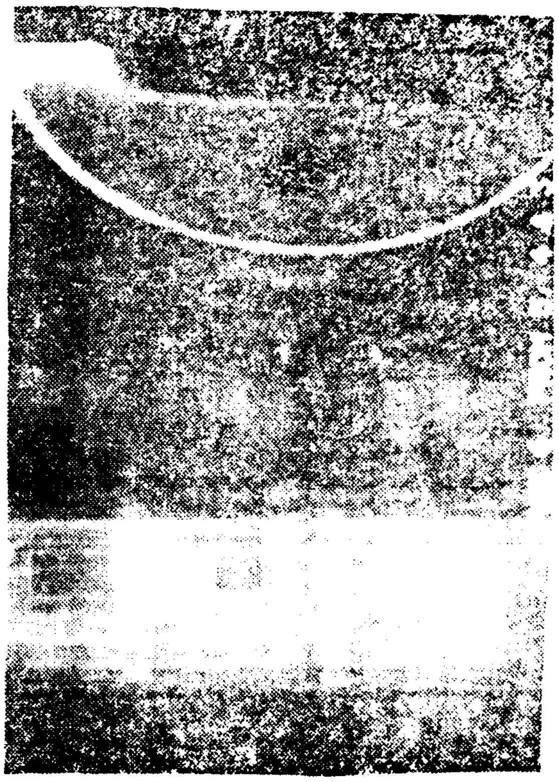
(f)

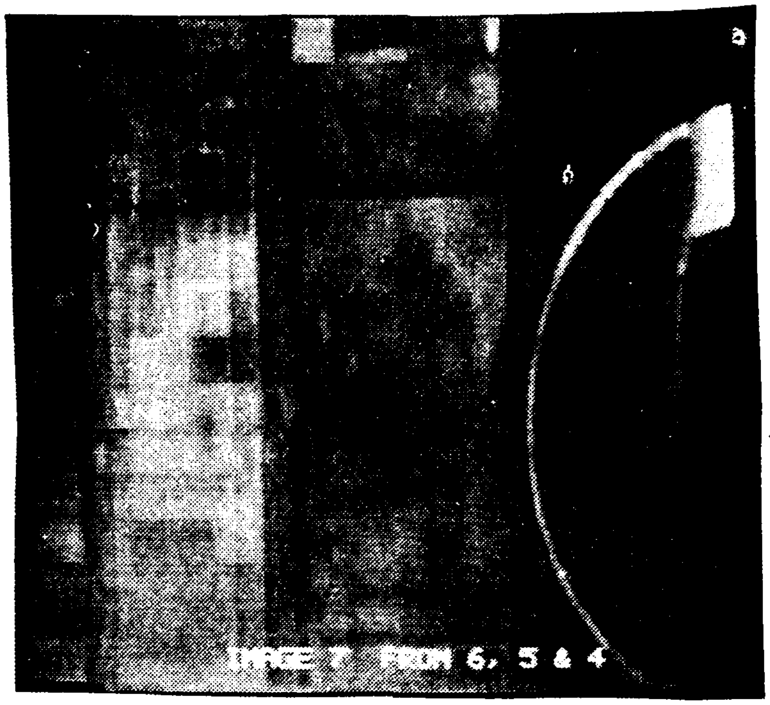
Figure 12. (Continued) Next-frame prediction via fixed predictor technique using las two frames received. The resulting images and the error between the original and the reconstructed frame represented by the signal-to-noise ratio (SNR).
(e) Image 7 from Images 6 and 5, SNR = 29.1 dB. (f) Image 8 from Images 7 and 6, SNR = 28.4 dB.

Figure 13. Next-frame prediction via fixed predictor technique using last three frames received. The resulting images and the error between the original and the reconstructed frame represented by the *signal-to-noise ratio* (SNR).
(a) Image 4 from Images 3, 2, and 1, SNR = 33.1 dB. (b) Image 5 from Images 4, 3, and 2, SNR = 33.5 dB. (c) Image 6 from Images 5, 4, and 3, SNR = 27.5 dB.

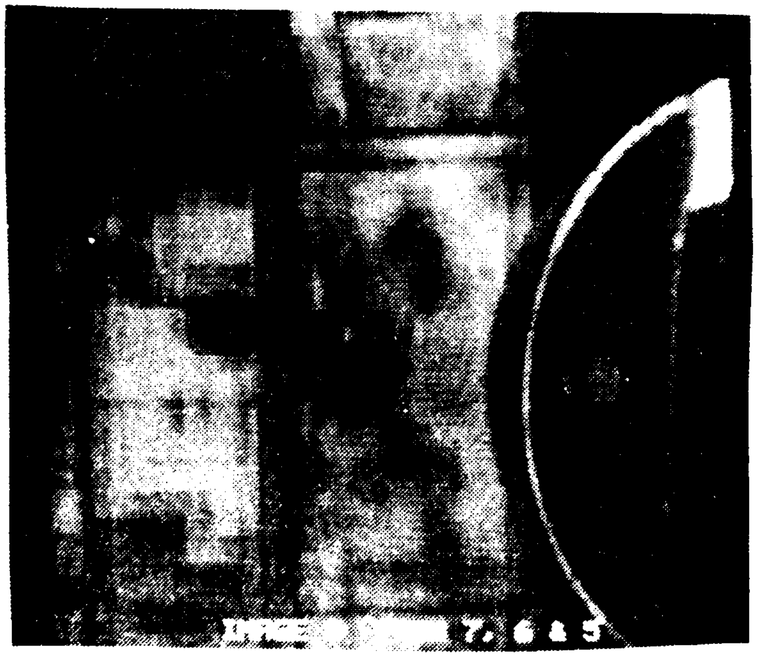

(b)


(a)


(c)



(d)



(e)

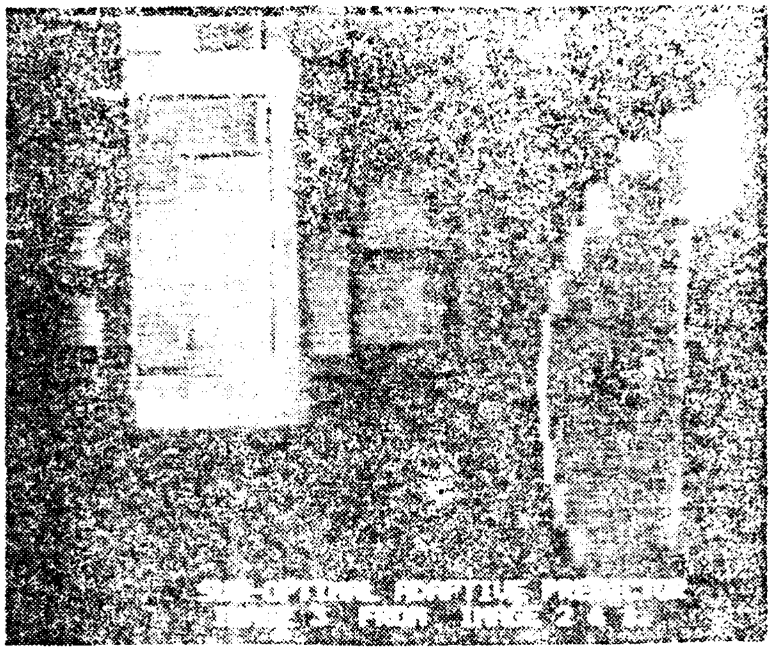
Figure 13. (Continued) Next-frame prediction via fixed predictor technique using last three frames received. The resulting images and the error between the original and the reconstructed frame represented by the signal-to-noise ratio (SNR).
(d) Image 7 from Images 6, 5, and 4, SNR = 28.5 dB. (e) Image 8 from Images 7, 6, and 5, SNR = 27.9 dB.

monitor, we can measure that distance to be exactly 32 pixels wide. This, however, does not make the image unacceptable.

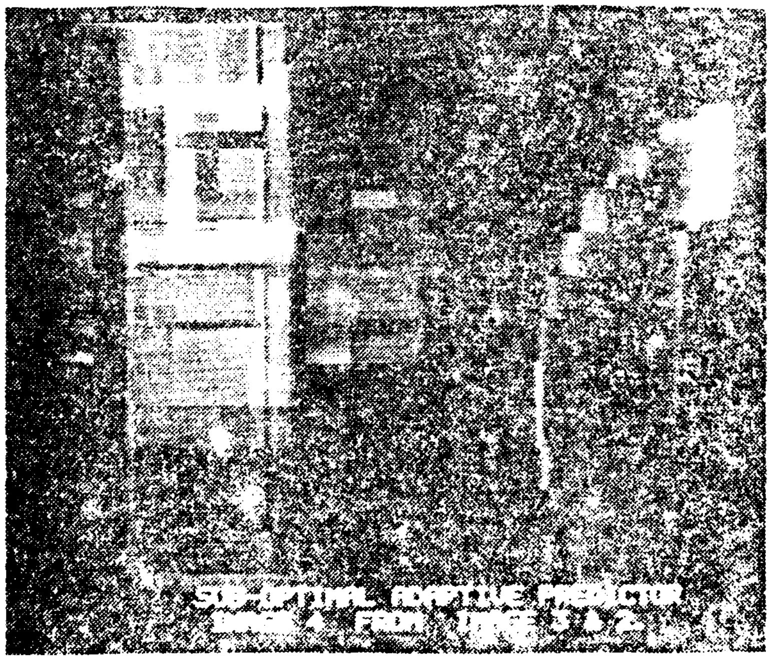
From an objective evaluation of the fixed predictor versus frame refresh, we can infer that the fixed predictor performs better than the frame refresh in almost all cases. However, the subjective quality of the reconstructed images is such that there does not seem to be a significant advantage of using this predictor with its attendant computations. The significant limitation of the fixed predictor is in those situations where the next frame has new and, therefore, unpredictable information. The structure of the model chosen in Chapter 3 does not allow us to compensate for such physical phenomena as tilting or pan, which introduce new information. This is obvious from the lower criterion values and poor quality of the estimates when predicting certain images compared to predicting others. For example, the estimate of image 6 from 5 is less accurate compared to the other estimates. When we apply the displacement-measurement algorithm described in Chapter 5 to the estimates, we find that improvements in SNR of the order of approximately 2 dB are achievable. The results are summarized in Table 9 for a set of cases. This is a significant gain since the results thus obtained are approximately 3 dB better than those available with the frame refresh technique. The displacement compensation procedure assumes that over a short sequence of frames, the interframe displacement can be assumed constant. Then, for the prediction, we would use instead of the actual frames, frames displaced by the same amount as the displacement between the preceding two frames. In this work, we used only actual frames. Here, it is appropriate to point out that an algorithm that measures frame displacement with fractional pixel accuracy is required to give an accurate estimate for the image when averaged over all its subblocks. Otherwise, the approximation over individual subblocks makes the overall estimate less accurate. In the specific application that we are considering, namely, that of satellite rendezvous and docking operation, the scene is characterized by very slow motion which would make the interframe displacement considerably smaller than what we see here. As mentioned before, the frames are not successive frames from the tape, but were deliberately chosen to be several frames apart. In either case, the algorithms are not very complex, hence an on-line movement-compensated prediction appears to be feasible and practical. This area requires further investigation as it is expected to give better visual results than what we have obtained from the fixed predictor.

Another conclusion that we can draw from an evaluation of the estimates, and the tables summarizing the criterion values, is that one-step ahead prediction almost always performs better than two- and three-step ahead predictions in that order. Where the interframe motion is small, e.g., between images 1 and 2, the prediction is more accurate than in the other cases (Figure 9). An evaluation of the results of the two- frame ahead prediction (Figure 10) shows a good picture with an attendant SNR value of 33.1 dB. Three-frame ahead prediction does not appear to be as good as the other two (Figure 11). A subjective evaluation of Figure 12 (next-frame prediction using last two frames received) shows that the prediction of image 3 from 2 and 1, image 7 from 6 and 5, and image 8 from 7 and 6 are the acceptable ones. This again appears to be a result of the interframe displacement. There also appears to be some duplication of feature points, more apparent in some cases than the others. This is because prediction using the last two frames received can be thought of as a projection of the last two frames onto the next frame. A change in the location of certain feature points between consecutive frames results in a shadow effect on the frame estimate. The results of the next frame prediction using the last three frames received is the worst in terms of the visual perception of the scene. The blurring which makes the target interpretation almost impossible is caused by that portion of the image which is nonstationary. This can be inferred from the fact that in that class, the acceptable estimates are those of image 8 (from 7, 6, and 5) and image 7 (from 6, 5, and 4). In the original frame sequence, the most significant interframe displacement is between images 2 and 3, which is why the other estimates seem so blurry. In almost all the cases, the SNR is approximately 30 dB.

The results of the suboptimal adaptive predictor are summarized in Table 8. The corresponding pictures are presented in Figure 14. From the objective evaluation, the suboptimal predictor matches the performance of the fixed predictor which, as we have seen, outperforms

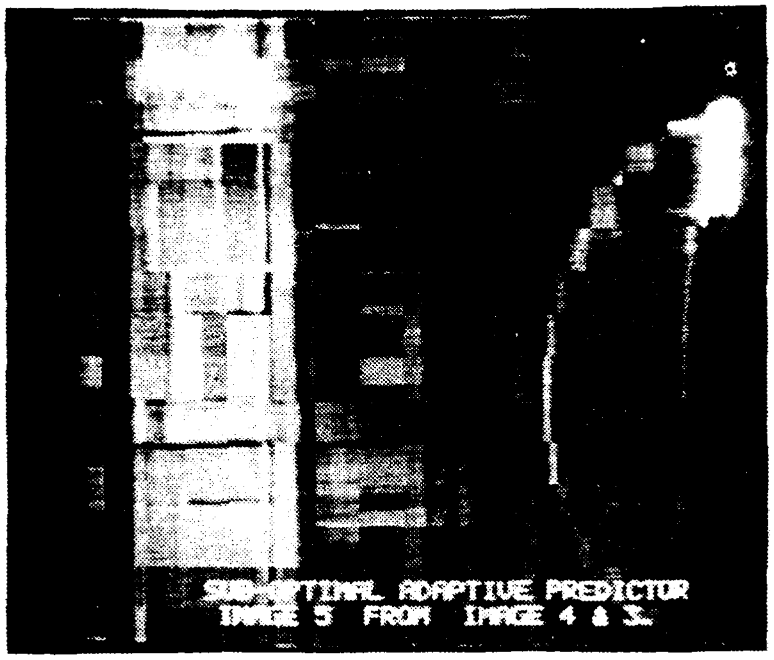


(a)

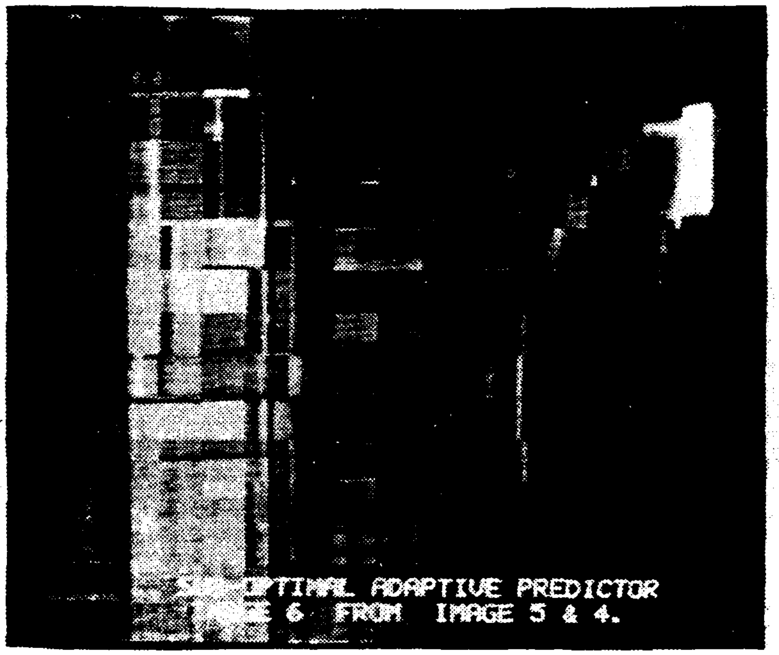


(b)

Figure 14 Next frame prediction via a suboptimal adaptive predictor technique using last two frames received. The resulting images and the error between the original and the reconstructed frame represented by the signal-to-noise ratio (SNR).
(a) Image 3 from Image 2 and 1. SNR = 33.2 dB.
(b) Image 4 from Image 3 and 2. SNR = 31.7 dB.



(c)



(d)

Figure 14. (Continued) Next-frame prediction via a suboptimal adaptive predictor technique using last two frames received. The resulting images and the error between the original and the reconstructed frame represented by the signal-to-noise ratio (SNR).
(c) Image 5 from Image 4 and 3, SNR = 32.1 dB.
(d) Image 6 from Image 5 and 4, SNR = 27.0 dB.

frame refresh. However, the approximation of the inverse transformation used in the derivation of the predictor introduces noise in the image recovery which is seen as the blurring effect. The only acceptable estimate in this category is that of image 3 based on images 2 and 1. When we display the original images 1, 2, and 3, we find that the relative displacement between 1 and 2 matches closely with that of 2 and 3. This suggests that in cases where the assumption of wide-sense stationarity is valid, the prediction is indeed feasible and of an acceptable quality. As described in Chapter 4, an investigation of the solution of singular matrices is in order since that is expected to improve on the results obtainable with a fixed predictor as well as a suboptimal adaptive predictor.

The aim of this work is to investigate if it is realistic to model the scene dynamics as a discrete-time linear state vector model, and estimate the model structure which is otherwise either completely unspecified or ill-defined. We do this by either exploiting some inherent properties of the images by trying to estimate the dynamics via an analysis of the available information. In either case, the subjective quality of the reconstructed estimates is such that the computations involved in deriving the predictors may not be justified. However, in some other cases, such as movement-compensated predictors, it may be possible to obtain far better results in which case the advantage over frame refresh may warrant the added computations. This area merits further investigation.

CHAPTER 7
CONCLUSIONS

The problem of dynamic estimation of unknown, time-varying parameters using a priori information is considered. The unknown parameters are the image pixel intensities of the next frames of a video sequence. Also, the dynamics of the model are not available and must be estimated. The measurable data are the pixel intensities of the preceding frames. We approach the problem by first representing the image dynamics as a linear state space vector model where each frame is represented as a state vector of dimension $N(= N_1 \times N_2)$, sampled at discrete instants. We then attempt to use the a priori information to estimate the dynamics of the model and to derive an on-line adaptive estimation of the unknown parameters. In the fixed predictor case, we use the inherent adjacent-pixel correlation of the image data for deriving the state vector model. The intent of the work is to determine if it is realistic to model the image dynamics via these approaches and if so, to investigate if it is possible to obtain a significant improvement over the frame refresh technique which requires no computations at all.

We find that it is feasible to do an on-line prediction of the image data and if certain conditions such as slow dynamics are satisfied, then it is possible to match the performance of the frame refresh technique according to the objective criteria. The conditions assumed are realistic in the context of teleoperator-controlled remote piloting application which is the main motivation for this work. It is also shown that in the case of slow dynamics, we can improve the criterion values by predicting the interframe displacement and compensating for it. However, from a subjective evaluation of the results of the fixed and adaptive predictors versus the frame refresh, we find that there is not a tremendous improvement over the frame refresh technique. The latter is attractive since it does not require any computations. We also find that a significant limitation of the fixed predictor approach is in those situations where the scene is characterized by rapid movement which introduces new information in the form of unmodeled dynamics. This indicates that the pixel correlation by itself may not be sufficient to model the image dynamics. In such cases, the use of an optimal adaptive predictor would be in order provided a computationally simple solution or approximation to the problem of inverting singular matrices could be found. The suboptimal predictor used in this case gives good criterion values. However, the matrices representing the image pixel intensities are seen to be extremely ill-conditioned making them singular. Even a generalized inverse solution fails to provide acceptable estimates. This is because the approximation used for the generalized inverse of the ill-conditioned matrices greatly degrades the visual quality of the images by effectively increasing the contribution of the noise which represents the unmodeled dynamics.

From the video sequence that was analyzed and the specific predictors used in this work, it appears that the improvement over frame refresh is not significant enough to warrant the computations required for these approaches; this does not preclude the possibility of obtaining significant improvement when using other approaches. Two areas that merit further investigation in this regard are as follows:

Movement-compensated on-line prediction appears to be a promising approach. Our work in Chapter 5 shows that it is possible to improve on the criterion values by estimating interframe displacement and compensating for it. It is expected that if one were to use the displaced frames instead of the actual frames for image data prediction and thus account for the interframe displacement, then the predicted frames would be more accurate than what one could get from the frame refresh. This is a topic for future research in this area.

Another area which merits further investigation is the problem of the ill-conditioned nature of the image processing matrices. Andrews and Patterson [19] and Huang [20] suggest an approach for solving this problem where a direct pseudoinverse gives unacceptable restored images due to the ill-conditioned nature of the matrices. This approach involves representing the image model as a separable space-variant model and consists of applying SVD to both row and column matrices separately. This is equivalent to decomposing the images into eigenimages and reconstructing the image by selectively discarding certain eigenimages. This area requires further investigation as an application of generalized inverses for image processing systems.

APPENDIX A

SUPPORTING SOFTWARE

A.1. Software for Fixed Predictor and Data Manipulation for D/A Conversion

```
      program pred
c
c PURPOSE:
c    This routine computes the next frame of the video
c    sequence using the last frame received and prepares
c    the frame estimate for D/A conversion by converting
c    it from real to binary data.
c
c ROUTINES CALLED: MTXMLT
c
c LOCAL VARIABLES, INCLUDING DIMENSIONS AND DESCRIPTION:
c       X(1:32,1:32), Y(1:32,1:32) - input matrices which hold
c           32x32 block of Image 1 and Image2 respectively
c           after converting integers to real numbers.
c       M(1:32,1:32), N(1:32,1:32) - input matrices which hold
c           32x32 block of Image 1 and Image2 respectively after
c           converting binary numbers to integers.
c       XEST(1:32,1:32) - matrix containing the estimate derived.
c       PHI(1:32,1:32) - matrix representing the fixed predictor.
c       Datal(512), Data2(512), Xsdata(512) - 512 byte arrays to
c           hold one record of imagel, image2 and the estimate
c           respectively.
c       Xdata(512) - integer array to hold the estimate data
c           after real to integer conversion.
c       XS1.dat - XS8.dat - files to hold the reconstructed
c           image as it is being created column- by- column.
c*****************************************************************
        implicit real*8 (a-h,o-z)
        implicit integer*4 (i-n)
        byte datal(512)
        byte data2 (512)
        byte xsdata (512)
        integer *2 xdata(512)
        integer *4 M(1:32,1:32)
        integer *4 N(1:32,1:32)
        real *8 PHI(1:32,1:32)
        real *8 X(1:32,1:32)
        real *8 XEST(1:32,1:32)
        real *8 Y(1:32,1:32)
        character *20 Imagel, Image2
c
        write (6,*) 'Type filename.typ for the first file.'
        read (5,1) Imagel
1       format (A)
        write (6,* ) 'Type filename.typ for the next file
       + of the sequence.'
        read (5,2) Image2
2       format (A)
c            Open all the units
```

```
      open (unit=11,file= Image1, status='old',form='unformatted',
     -       recordtype='fixed', recl=128)
      open (unit=13,file= Image2, status='old',form='unformatted',
     +       recordtype='fixed', recl=128)
      open (unit=12,file='xs1.dat',status='new',form='unformatted',
     +       recordtype='fixed', recl=8)
      open (unit=14,file='xs2.dat',status='new',form='unformatted',
     +       recordtype='fixed', recl=16)
      open (unit=15,file='xs3.dat',status='new',form='unformatted',
     +       recordtype='fixed', recl=24)
      open (unit=16,file='xs4.dat',status='new',form='unformatted',
     +       recordtype='fixed', recl=32)
      open (unit=17,file='xs5.dat',status='new',form='unformatted',
     +       recordtype='fixed', recl=40)
      open (unit=18,file='xs6.dat',status='new',form='unformatted',
     +       recordtype='fixed', recl=48)
      open (unit=19,file='xs7.dat',status='new',form='unformatted',
     +       recordtype='fixed', recl=56)
      open (unit=20,file='xs8.dat',status='new',form='unformatted',
     +       recordtype='fixed', recl=64)
c                Initialize the phi-matrix
      do k=1,32
      do l=1,32
      phi (k,l) = 0.0
      end do
      end do
cSet the phi- matrix
      do l=1,32,31
      do k=1,4
      phi(l,l) = 1.0/ 4.62
      if ((i+k) .le. 32) then
      phi(l,l+k)= (0.96 ** k) / 4.62
      end if
      if ((l-k) .ge. 1 ) then
      phi(l,l-k) = (0.96 ** k) /4.62
      end if
      end do
      end do
c
      do j=2,31,29
      do k=1,4
      phi(j,j) = 1.0/ 5.58
      if ((j+k) .le. 32) then
      phi(j,j+k)= (0.96 ** k) / 5.58
      end if
      if ((j-k) .ge. 1 ) then
      phi(j,j-k) = (0.96 ** k) / 5.58
      end if
      end do
      end do
```

```
        do i=3,30,27
        do k=1,4
        phi(i,i) = 1.0/ 6.5
        if ((i+k) .le. 32) then
        phi(i,i+k)= (0.96 ** k) / 6.5
        end if
        if ((i-k) .ge. 1 ) then
        phi(i,i-k) = (0.96 ** k) / 6.5
        end if
        end do
        end do
        do j=4,29,25
        do k=1,4
        phi(j,j) = 1.0/ 7.38
        if ((j+k) .le. 32) then
        phi(j,j+k)= (0.96 ** k) / 7.38
        end if
        if ((j-k) .ge. 1 ) then
        phi(j,j-k) = (0.96 ** k) / 7.38
        end if
        end do
        end do
        do i=5,28
        do k=1,4
        phi(i,i) = 1.0/8.23
        if ((i+k) .le. 32) then
        phi (i,i+k) =  (0.96 **k) /8.23
        end if
        if ((i-k) .ge. 1 ) then
        phi (i,i-k) = (0.96 ** k) / 8.23
        end if
        end do
        end do
c               Read Image1
        npin = 512
        npout = 32
        nrout = 32
c                Set the initial values for performance measures
           sum1 = 0.0
           sum2 = 0.0
           sum3 = 0.0
           sum4 = 0.0
           sum5 = 0.0
           sum6 = 0.0
           sum 7 = 0.0
           sum8 = 0.0
           snr = 0.0
c               Start processing column 1: 8
        do 10 npskip = 0,224,32
c               Start processing a 32x32 block in the first column
          do 20 k=1,8
```

```
c             Read next nrout (=32 in this case) records!
       do i = 1, nrout
         read (11) (datal(j), j=1,npin)
         do j = 1, npout
         M(i,j) = jzext( datal(npskip+j) )
         X(i,j) = dflotj( M(i,j) )
         end do
      end do
c             Cross-check.
      write (6,3) K,X(1,1)
3     format (/,1x,'BLOCK #',I4,', X(1,1)=',F9.3)
      call mtxmlt (phi,32,32,X,32,32,XEST)
c     Start writing a 32x32 block at a time, i.e.,
c     npskip+1 : npskip+32 after doing real to integer
c     conversion (integer data written to binary files!)
      do kp=1,32
      if (npskip .eq. 0) then
      go to 303
      else if (npskip .eq. 32) then
      read (12) (xsdata(j), j = 1,npskip)
      else if (npskip .eq. 64 ) then
      read (14) (xsdata(j), j=1,npskip)
      else if (npskip .eq. 96) then
      read (15) (xsdata(j), j = 1,npskip)
      else if (npskip .eq. 128) then
      read (16) (xsdata(j), j = 1,npskip)
      else if (npskip .eq. 160) then
      read (17) (xsdata(j), j = 1,npskip)
      else if (npskip .eq. 192) then
      read (18) (xsdata(j), j = 1,npskip)
      else if (npskip .eq. 224) then
      read (19) (xsdata(j), j = 1,npskip)
      end if
c
303   do l=1,32
      xdata(npskip+l) = iidnnt( xest(kp,l))
      if (xdata(npskip+l) .ge. 127) then
      xdata(npskip+l) = (xdata(npskip+l)) - 255
      end if
      xsdata (npskip+l) = xdata(npskip+l)
      end do
      if (npskip .eq. 0) then
      write (12) ( xsdata(j), j=1,(npskip+32) )
      else if (npskip .eq. 32) then
      write (14) ( xsdata(j), j=1,(npskip+32) )
      else if (npskip .eq. 64) then
      write (15) ( xsdata(j), j=1,(npskip+32) )
      else if (npskip .eq. 96) then
      write (16) ( xsdata(j), j=1,(npskip+32) )
      else if (npskip .eq. 128) then
      write (17) ( xsdata(j), j=1,(npskip+32) )
```

```
        else if (npskip .eq. 160) then
        write (18) ( xsdata(j), j=1,(npskip+32) )
        else if (npskip .eq. 192) then
        write (19) ( xsdata(j), j=1,(npskip+32) )
        else if (npskip .eq. 224) then
        write (20) ( xsdata(j), j=1,(npskip+32) )
        end if
        end do
c              Cross-check.
        write (6,4) xest(1,1)
4       format (1x,'XEST(1,1) =',F9.3 )
c             Now read IMAGE2.DAT
c             Read nrout (=32 in this case) records at a time!
        do i = 1, nrout
           read (13) (data2(j), j=1,npin)
           do j = 1, npout
            N(i,j) = jzext( data2(npskip+j) )
            Y(i,j) = dflotj( N(i,j) )
           end do
         end do
c             Cross-check.
         write (6,5) Y(1,1)
5        format (1X,'Y(1,1)=',F9.3)
c            Compute mabse, mse, % nmse and SNR.
        do i = 1,32
          do j=1,32
             sum1 = sum1 + (dabs (Y(i,j) -Xest(i,j)))
             sum2 = sum2 + ( Y(i,j) -Xest(i,j) ) ** 2.0
             sum3 = sum3 + (Y(i,j) ** 2.0)
          end do
         end do
c          Close the loop for each 32x32 block (total 8).
20        continue
c          After 8 iterations, close the loop for individual
c          columns (total 8), i.e., npskip+1: npskip+32, and
c          print the performance measures for each column
             sum4 = sum2 / sum3
             sum1 = sum1 / (1024 * 8)
             sum2 = sum2 / (1024 * 8)
             sum3 = sum3 / (1024 *300)
          write (6,7) sum1, sum2, sum4, sum3
7       format (1x,'MABSE=',E9.3,1x,'MSE=',E9.3,1x,'NMSE=',E9.3,
     +   1x, 'MEAN  IMAGE  VAR. /100 =',E12.6)
c          Reinitialize the performance measures for each
c          column iteration
          sum5= sum5 + sum1
          sum6 = sum6 +sum2
          sum7 = sum7 + sum3
          sum1 = 0.0
          sum2 = 0.0
        sum3 = 0.0
        sum 4 = 0.0
c             Close units 11,12,13 and reopen in order to
c             start reading records from the 1st record.
        rewind (11)
        rewind (13)
       if (npskip .eq. 0) then
       close (12)
       open (unit=12,file='xs1.dat',status='old',form =
     + 'unformatted', readonly, recordtype='fixed', recl=8)
       else if (npskip .eq. 32) then
       close (14)
       open (unit=14,file='xs2.dat',status='old',form =
     + 'unformatted',readonly, recordtype='fixed', recl=16)
       else if (npskip .eq. 64) then
       close (15)
       open (unit=15,file='xs3.dat',status='old', form=
     + 'unformatted',readonly, recordtype='fixed', recl=24)
       else if (npskip .eq. 96) then
       close (16)
       open (unit=16,file='xs4.dat',status='old',form =
     + 'unformatted',readonly, recordtype='fixed', recl=32)
       else if (npskip .eq. 128) then
       close (17)
       open (unit=17,file='xs5.dat',status='old',form =
     + 'unformatted',readonly, recordtype='fixed', recl=40)
       else if (npskip .eq. 160) then
       close (18)
       open (unit=18,file='xs6.dat',status='old',form =
     + 'unformatted',readonly, recordtype='fixed', recl=48)
       else if (npskip .eq. 192) then
       close (19)
       open (unit=19,file='xs7.dat',status='old',form =
     + 'unformatted',readonly, recordtype='fixed', recl=56)
       else if (npskip .eq. 224) then
       close (20)
       end if
10      continue
        write (6,11) Image2, Image1, Npskip
11     format (1x,'THIS PROGRAM PREDICTS',A,' FROM ', A ,
     + ' FOR A  256X32  BLOCK AFTER DELETING',I4,' COLUMNS.',
     + /,'FINISHED PROCESSING THE FIRST 256X256 IMAGE')
        SUM5 = SUM5 / 8.
        SUM6 = SUM6 / 8.
        SUM 7= SUM7 /8.
        SUM8 = SUM6/SUM7
        SNR = SNR + ((255 ** 2.0)/ SUM6)
        SNR = 10.0 * dlog10 (snr)
        write (6,12) sum5, sum6, sum7
12     format (1x,'TOTAL MABSE=',E9.3,1x,'TOTAL MSE=',E9.3,1x,
     + 'MEAN  IMAGE  VAR./100 OVER THE ENTIRE BLOCK =',E12.6)
c
        write (6, 13) sum8,snr
13     format (1X, '% TOTAL NMSE= ,E9.3,1X,'SNR=',F6.3 )
        end
```

```
      subroutine mtxmlt(mtxa,nrowa,ncola,mtxb,nrowb,ncolb,mtxc)
c
c  PURPOSE:
c    This subroutine computes the matrix product of two matrices.
c  CALLING SEQUENCE:
c     CALL MTXMLT (MTXA,NROWA,NCOLA,MTXB,NROWB,NCOLB,MTXC)
c     MTXA: (input) first matrix of product, maximum dimension
c              MTXA(1:32,1:32).
c     NROWA: (input) number of rows in MTXA.
c     NCOLA: (input) number of columns in MTXA.
c     MTXB: (input) second matrix of product, maximum dimension
c              MTXB(1:32,1:32).
c     NROWB: (input) number of rows in MTXB.
c     NCOLB: (input) number of columns in MTXB.
c     MTXC: (output) resultant product matrix of MTXA times
c             MTXB, maximum dimension MTXC(1:32,1:32).
c  ROUTINES CALLED:  none.
c  LIMITATIONS:
c   The maximum dimension of the input and output matrices
c   is 32x32.
c
c  ******************* END OF PREFACE ******************************
      implicit real*8 (a-h,o-z)
      implicit integer*4 (i-n)
      real*8 mtxa(1:nrowa,1:ncola), mtxb(1:nrowb,1:ncolb)
      real*8 mtxc(1:nrowa,1:ncolb)
      if ((ncola-nrowb).eq.0) then
         if ((nrowa.le.32).and.(nrowb.le.32).and.(ncola.le.32).and.
     >       (ncolb.le.32)) then
            do 100 i = 1,nrowa
               do 200 j = 1,ncolb
                  mtxc(i,j) = 0.0
                  do 300 k = 1,ncola
                     mtxc(i,j) = mtxc(i,j) + mtxa(i,k)*mtxb(k,j)
 300              continue
 200           continue
 100        continue
         else
            write(20,9000)
         end if
      else
         write (20,9010) ncola, nrowb
         write (20,9020)
      end if
 9000 format (/,' dimension of one or both MTXMLT input',
     >          ' matrices are larger than 32x32.')
 9010 format (/,'   mtxa has',i2,' columns;  mtxb has ',i2,
     >          ' rows.')
 9020 format (/,' matrix multiplication is not defined in',
     >          ' this situation.')
      return
      end
```

A.2. Software for Interframe Displacement Estimation

```
      program motion
c
c PURPOSE:  This routine computes the interframe  displacement
c   between 2 frames of a video sequence using a minimum mean
c   square error criterion accoding to the algorithm by Jain
c   and Jain.
c LOCAL VARIABLES INCLUDING DIMENSIONS AND DESCRIPTION:
c
c   X(1:32,1:32), Y(1:32,1:32) - input matrices to hold 32x32
c    blocks of images 1 & 2, respectively after integer to real
c    conversion.
c   M(1:32,1:32), N(1:32,1:32) - input matrices to hold the
c    above data after byte to integer conversion.
c   Bdata(512), Data(512) -  512 byte arrays to hold one record
c    of image1 and image 2,respectively, at any given time.
c
c   Sum(1:5) - array to hold the variances for a 5-point search.
c   Sum2(1:9) - array to hold the variances for the final
c    9-point search.
c   Summ, Summ2 - working variables for the minimum variance
c    over 5- and 9=point search, respectively.
c   P, PK, NN - working variables.
c   Q, L and  TQ, TL - working variables for the x,y displacement
c    over a block, and over the entire image, respectively.
c
c   VALI(1:5), VALJ(1:5) and  VALI2(1:9), VALJ2(1:9) - arrays to
c    hold the x,y values for the 5- and 9-point search,
c    respectively.
c *****************************************************************
c
      implicit real*8 (a-h,o-z)
      implicit integer*4 (i-n)
      byte bdata(512)
      byte data (512)
      character *63 Infile1, Infile2
      real *8 sum(1:5), sum2(1:9)
      real *8 summ, summ2, tsum
      integer *4 q, p, pk, nn, tq, tl
      real *8 X(1:48,1:48)
      real *8 Y(1:64,1:64)
      integer *4 vali(1:5)
      integer *4 valj(1:5)
      integer *4 vali2(1:9)
      integer *4 valj2(1:9)
      integer *4 M(1:48,1:48)
      integer *4 N(1:64,1:64)
c
      write (6,*) 'Type Filename.type for the estimate file. '
      Read (5,1) Infile1
      write (6,*) 'What is it the estimate of?  Type "Filename.typ"
     + for the estimate file. '
      Read (5,1) Infile2
1     format (A)
```

```
c               Initialize variables
      npin1 = 256
      npout1 = 32
      nrout1 = 32
      npin2 = 512
      tq=0
      t1=0
      tsum = 0.0
c
      do 20 npskip1 = 0,224,32
         do 10 nrskip1 = 0,224,32
c
c          Open all the units
     open (unit=11, file= infile1, status='old',
    +    form='unformatted', recordtype='fixed', recl=64)
     open (unit=12, file= infile2, status='old',
    +   form='unformatted', recordtype='fixed', recl=128)
c
      npout2 = 0
      nrout2 = 0
c        For each 32x32 pixel sub-block in the 1st image,
c        pick a 64x64 sub-block in the 2nd image centered
c        at the same spatial location. In the corners
c        we would have only 48 pixels along one or both axes.
c
     If (nrskip1 .ge. 32) then
     nrskip2 = nrskip1 -16
     else if (nrskip1 .eq. 0) then
     nrskip2 = nrskip1
     end if
c
     If (npskip1 .ge. 32) then
     npskip2 = npskip1 -16
     else if (npskip1 .eq. 0) then
     npskip2 = npskip1
     end if
c
     If ((nrskip1 .eq. 0) .or. (nrskip1 .eq. 224)) then
     nrout2 = 48
     else
     nrout2 = 64
     end if
c
     If ((npskip1 .eq. 0) .or. (npskip1 .eq. 224)) then
     npout2 = 48
     else
     npout2 = 64
     end if
c
     do i=1,48
       do j=1,48
       X(i,j) = 0.0
```

```
          M(i,j) = 0
          end do
        end do
        do k=1,64
           do l=1,64
           Y(k,l) = 0.0
           N(k,l) =0
           end do
        end do
        do l = 1,5
        vali(l) = 0
        valj(l) = 0
        end do
        do p = 1,9
        vali2(p) = 0
        valj2(p) = 0
        end do
        p=0
        k=0
        pk=0
c               Read Infile1.
c               First skip nrskip1 records!
        do i =1,nrskip1
        read(11) bdata(1)
        end do
c               Read next nrout1 records!
        do i = 17, 48
           read (11) (bdata(j), j=1,npin1)
           k = 1
           do j = 17, 48
           M(i,j) = jzext( bdata(npskip1+k) )
           X(i,j) = dflotj( M(i,j) )
           k = k +1
           end do
        end do
c               Cross-check.
        If (nrskip1 .eq. 224) then
        write (6,890) NPSKIP1, NRSKIP1, X(17,17)
890     format (1x,'NPSKIP1 =',I4,',NRSKIP1 =',I4, ',X(1,1) =',F9.3)
        end if
c               Now read Infile2
c               First skip nrskip2 records!
        do i =1, nrskip2
        read(12) data(1)
        end do
c               Read next nrout2 records!
        If ( nrskip1 .eq. 0) then
        ni1 = 17
        else
        ni1 = 1
        end if
        ni2 = ni1 + nrout2 -1
```

```
      If ( npskipl .eq. 0) then
      njl = 17
      else
      njl = 1
      end if
      nj2 = njl + npout2 -1
     -do i = nil, ni2
         read (12) (data(j), j=1,npin2)
         k = 1
         do j = njl, nj2
          N(i,j) = jzext( data(npskip2+k) )
          Y(i,j) = dflotj( N(i,j) )
          k = k +1
         end do
       end do
c                 Cross-check.
       If ((npskipl .eq. 224) .and. (nrskipl .eq. 224)) then
       write (6,891) NPSKIP2, NRSKIP2, Y(nil, njl)
891   format (1x,'NPSKIP2 =',I4,',NRSKIP2 =',I4,',Y(1,1) =',F9.3)
       end if
c               Compute  the variances for the 5-point search.
c
       nn = 8
       q = 0
       l = 0
3000   continue
       vali(1) = 0
       valj(1) = 0
       vali(2) = 1 * nn
       valj(2) = 0
       vali(3) = 0
       valj(3) = 1 * nn
       vali(4) = -1 * nn
       valj(4) = 0
       vali(5) = 0
       valj(5) = -1 * nn
c
2020   continue
       do 2000 p =1,5
        sum (p) = 0.0
        k = 0
        pk = 0
        k= q + vali(p)
        pk = l + valj(p)
c            Variance = infinity, if 16 < i,j < -16, i.e.,
c              maximum pixel displacement is 16 pixels.
        If (( k .gt. 16) .or. ( k .lt. -16) .or.
     + (pk .gt. 16) .or. (pk .lt. -16) ) then
        sum(p) = 1000000.0
        else
         do 200 ml = 17,48
         do 100 nl = 17,48
```

```
          sum(p) = sum(p) + ( X(ml,nl) - Y(ml+k,nl+pk) ) ** 2
100       continue
200       continue
          sum(p) = sum(p) / (32**2)
         end if
2000    end do
c              Compute the minimum variance
2005     continue
         summ = 0.0
         summ = SNGL( DMIN1( dble(sum(1)), dble(sum(2)), dble(sum(3)),
     +   dble(sum(4)), dble(sum(5)) ) )
c
c             Find the minimum.
         If (summ .eq. sum(1)) then
         pq=1
         else if (summ .eq. sum(2)) then
         pq =2
         else if (summ .eq. sum(3)) then
         pq=3
         else if (summ .eq. sum(4)) then
         pq=4
         else if (summ .eq. sum(5)) then
         pq=5
         end if
         q = q + vali(pq)
         l = l + valj(pq)
c
         If (pq .eq. 1) then
         go to 2010
         else if (pq .gt. 1) then
         go to 2020
         end if
2010     continue
         nn = nn/2
         If ( nn .eq. 1) then
         go to 4000
         else if ( nn .gt. 1) then
         go to 3000
         end if
4000    continue
        vali2(1) = 0
        valj2(1) = 0
        vali2(2) = 1
        valj2(2) = 0
        vali2(3) = 1
        valj2(3) = 1
        vali2(4) = 0
        valj2(4) = 1
        vali2(5) = -1
        valj2(5) = 1
        vali2(6) = -1
        valj2(6) = 0
```

```
         vali2(7) = -1
         valj2(7) = -1
         vali2(8) = 0
         valj2(8) = -1
         vali2(9) = 1
         valj2(9) = -1
c
c            Compute the variances for the final 9-point search,
c            i.e., over -1 (=,<) i,j (=,<) 1.
         do 5000 p =1,9
          sum2(p) = 0.0
          k = 0
          pk = 0
          k= q + vali2(p)
          pk = 1 + valj2(p)
          If (( k .gt. 16) .or. ( k .lt. -16) .or.
     +    (pk .gt. 16) .or. (pk .lt. -16) ) then
          sum2(p) = 1000000.0
          else
           do 500 ml = 17,48
           do 400 nl = 17,48
           sum2(p) = sum2(p) + ( X(ml,nl) - Y(ml+k,nl+pk) ) ** 2
400        continue
500        continue
           sum2(p) = sum2(p) / (32**2)
          end if
5000     end do
c               Compute the minimum variance
3005       summ2 = 0.0
           summ2 = SNGL( DMIN1( dble(sum2(1)), dble(sum2(2)),
     +     dble(sum2(3)), dble(sum2(4)), dble(sum2(5)), dble(sum2(6)),
     +     dble(sum2(7)), dble(sum2(8)), dble(sum2(9)) ))
c
c               Find the minimum.
          If (summ2 .eq. sum2(1)) then
          pq=1
          else if (summ2 .eq. sum2(2)) then
          pq =2
          else if (summ2 .eq. sum2(3)) then
          pq=3
          else if (summ2 .eq. sum2(4)) then
          pq=4
          else if (summ2 .eq. sum2(5)) then
          pq=5
          elseif (summ2 .eq. sum2(6)) then
          pq=6
          else if (summ2 .eq. sum2(7)) then
          pq =7
          else if (summ2 .eq. sum2(8)) then
```

```
              pq=8
              else if (summ2 .eq. sum2(9)) then
              pq=9
              end if
              q = q + vali2(pq)
              l = l + valj2(pq)
c                  Again, limit the displacement to x,y (<,=) 16.
c
             If ( (q .lt. (-1*16)) .or. (q .gt. 16) .or. (l .gt. 16)
     +       .or. (l .lt. (-1*16)) )then
             sum(pq) = 1000000.0
             q = q - vali(pq)
             l = l - valj(pq)
             go to 3005
             end if
c
              tq = tq + q
              tl = tl + l
              tsum = tsum + summ2
c
6001         close (11)
             close (12)
10           END DO
c
              write (6,794) npskipl, nrskipl,nn,tq,tl,tsum
794           format (1x,'NPSKIP1=',I4,' ,NRSKIP1=',I4,' ,nn=',I4,/,
     +        ' TOTAL (SO FAR) DMD DIMENSIONS: q =',I4,' ,l =',I4,/,
     +        ' ,MIN (SO FAR) D(q,l) =',F10.3)
20            END DO
c
              write (6,993) infile1, infile2
993           format (/,'THIS PROGRAM COMPUTES THE MINIMUM VARIANCE
     +        BETWEEN',A,'AND', A,'FOR THE 1ST 256x256 SUB-IMAGE.')
c
              write (6,804) tq,tl,tsum
804           format (/,' TOTAL DMD DIMENSIONS: q =',I4,' ,l =',I4,
     +        ' ,MIN AVG VAR =',F10.3)
c
             end
```

APPENDIX B

NUMERICAL VALUES ASSOCIATED WITH THE IMAGE PROCESSING PROBLEM

B.1. Non-Zero Elements of A-Matrix

A (1, 1) =	0.216	A (1, 2) =	0.208	A (1, 3) =	0.199
A (1, 4) =	0.192	A (1, 5) =	0.184	A (2, 1) =	0.172
A (2, 2) =	0.179	A (2, 3) =	0.172	A (2, 4) =	0.165
A (2, 5) =	0.159	A (2, 6) =	0.152	A (3, 1) =	0.142
A (3, 2) =	0.148	A (3, 3) =	0.154	A (3, 4) =	0.148
A (3, 5) =	0.142	A (3, 6) =	0.136	A (3, 7) =	0.131
A (4, 1) =	0.120	A (4, 2) =	0.125	A (4, 3) =	0.130
A (4, 4) =	0.136	A (4, 5) =	0.130	A (4, 6) =	0.125
A (4, 7) =	0.120	A (4, 8) =	0.115	A (5, 1) =	0.103
A (5, 2) =	0.108	A (5, 3) =	0.112	A (5, 4) =	0.117
A (5, 5) =	0.122	A (5, 6) =	0.117	A (5, 7) =	0.112
A (5, 8) =	0.108	A (5, 9) =	0.103	A (6, 2) =	0.103
A (6, 3) =	0.108	A (6, 4) =	0.112	A (6, 5) =	0.117
A (6, 6) =	0.122	A (6, 7) =	0.117	A (6, 8) =	0.112
A (6, 9) =	0.108	A (6,10) =	0.103	A (7, 3) =	0.103
A (7, 4) =	0.108	A (7, 5) =	0.112	A (7, 6) =	0.117
A (7, 7) =	0.122	A (7, 8) =	0.117	A (7, 9) =	0.112
A (7,10) =	0.108	A (7,11) =	0.103	A (8, 4) =	0.103
A (8, 5) =	0.108	A (8, 6) =	0.112	A (8, 7) =	0.117
A (8, 8) =	0.122	A (8, 9) =	0.117	A (8,10) =	0.112
A (8,11) =	0.108	A (8,12) =	0.103	A (9, 5) =	0.103
A (9, 6) =	0.108	A (9, 7) =	0.112	A (9, 8) =	0.117
A (9, 9) =	0.122	A (9,10) =	0.117	A (9,11) =	0.112
A (9,12) =	0.108	A (9,13) =	0.103	A (10, 6) =	0.103
A (10, 7) =	0.108	A (10, 8) =	0.112	A (10, 9) =	0.117
A (10,10) =	0.122	A (10,11) =	0.117	A (10,12) =	0.112
A (10,13) =	0.108	A (10,14) =	0.103	A (11, 7) =	0.103
A (11, 8) =	0.108	A (11, 9) =	0.112	A (11,10) =	0.117
A (11,11) =	0.122	A (11,12) =	0.117	A (11,13) =	0.112
A (11,14) =	0.108	A (11,15) =	0.103	A (12, 8) =	0.103
A (12, 9) =	0.108	A (12,10) =	0.112	A (12,11) =	0.117
A (12,12) =	0.122	A (12,13) =	0.117	A (12,14) =	0.112
A (12,15) =	0.108	A (12,16) =	0.103	A (13, 9) =	0.103
A (13,10) =	0.108	A (13,11) =	0.112	A (13,12) =	0.117
A (13,13) =	0.122	A (13,14) =	0.117	A (13,15) =	0.112
A (13,16) =	0.108	A (13,17) =	0.103	A (14,10) =	0.103
A (14,11) =	0.108	A (14,12) =	0.112	A (14,13) =	0.117
A (14,14) =	0.122	A (14,15) =	0.117	A (14,16) =	0.112
A (14,17) =	0.108	A (14,18) =	0.103	A (15,11) =	0.103
A (15,12) =	0.108	A (15,13) =	0.112	A (15,14) =	0.117
A (15,15) =	0.122	A (15,16) =	0.117	A (15,17) =	0.112
A (15,18) =	0.108	A (15,19) =	0.103	A (16,12) =	0.103
A (16,13) =	0.108	A (16,14) =	0.112	A (16,15) =	0.117
A (16,16) =	0.122	A (16,17) =	0.117	A (16,18) =	0.112
A (16,19) =	0.108	A (16,20) =	0.103	A (17,13) =	0.103
A (17,14) =	0.108	A (17,15) =	0.112	A (17,16) =	0.117
A (17,17) =	0.122	A (17,18) =	0.117	A (17,19) =	0.112
A (17,20) =	0.108	A (17,21) =	0.103	A (18,14) =	0.103

```
A (18,15) =  0.108   A (18,16) =  0.112   A (18,17) =  0.117
A (18,18) =  0.122   A (18,19) =  0.117   A (18,20) =  0.112
A (18,21) =  0.108   A (18,22) =  0.103   A (19,15) =  0.103
A (19,16) =  0.108   A (19,17) =  0.112   A (19,18) =  0.117
A (19,19) =  0.122   A (19,20) =  0.117   A (19,21) =  0.112
A (19,22) =  0.108   A (19,23) =  0.103   A (20,16) =  0.103
A (20,17) =  0.108   A (20,18) =  0.112   A (20,19) =  0.117
A (20,20) =  0.122   A (20,21) =  0.117   A (20,22) =  0.112
A (20,23) =  0.108   A (20,24) =  0.103   A (21,17) =  0.103
A (21,18) =  0.108   A (21,19) =  0.112   A (21,20) =  0.117
A (21,21) =  0.122   A (21,22) =  0.117   A (21,23) =  0.112
A (21,24) =  0.108   A (21,25) =  0.103   A (22,18) =  0.103
A (22,19) =  0.108   A (22,20) =  0.112   A (22,21) =  0.117
A (22,22) =  0.122   A (22,23) =  0.117   A (22,24) =  0.112
A (22,25) =  0.108   A (22,26) =  0.103   A (23,19) =  0.103
A (23,20) =  0.108   A (23,21) =  0.112   A (23,22) =  0.117
A (23,23) =  0.122   A (23,24) =  0.117   A (23,25) =  0.112
A (23,26) =  0.108   A (23,27) =  0.103   A (24,20) =  0.103
A (24,21) =  0.108   A (24,22) =  0.112   A (24,23) =  0.117
A (24,24) =  0.122   A (24,25) =  0.117   A (24,26) =  0.112
A (24,27) =  0.108   A (24,28) =  0.103   A (25,21) =  0.103
A (25,22) =  0.108   A (25,23) =  0.112   A (25,24) =  0.117
A (25,25) =  0.122   A (25,26) =  0.117   A (25,27) =  0.112
A (25,28) =  0.108   A (25,29) =  0.103   A (26,22) =  0.103
A (26,23) =  0.108   A (26,24) =  0.112   A (26,25) =  0.117
A (26,26) =  0.122   A (26,27) =  0.117   A (26,28) =  0.112
A (26,29) =  0.108   A (26,30) =  0.103   A (27,23) =  0.103
A (27,24) =  0.108   A (27,25) =  0.112   A (27,26) =  0.117
A (27,27) =  0.122   A (27,28) =  0.117   A (27,29) =  0.112
A (27,30) =  0.108   A (27,31) =  0.103   A (28,24) =  0.103
A (28,25) =  0.108   A (28,26) =  0.112   A (28,27) =  0.117
A (28,28) =  0.122   A (28,29) =  0.117   A (28,30) =  0.112
A (28,31) =  0.108   A (28,32) =  0.103   A (29,25) =  0.115
A (29,26) =  0.120   A (29,27) =  0.125   A (29,28) =  0.130
A (29,29) =  0.136   A (29,30) =  0.130   A (29,31) =  0.125
A (29,32) =  0.120   A (30,26) =  0.131   A (30,27) =  0.136
A (30,28) =  0.142   A (30,29) =  0.148   A (30,30) =  0.154
A (30,31) =  0.148   A (30,32) =  0.142   A (31,27) =  0.152
A (31,28) =  0.159   A (31,29) =  0.165   A (31,30) =  0.172
A (31,31) =  0.179   A (31,32) =  0.172   A (32,28) =  0.184
A (32,29) =  0.192   A (32,30) =  0.199   A (32,31) =  0.208
A (32,32) =  0.216
```

REFERENCES

[1] A. J. Seyler, "The coding of visual signals to reduce channel-capacity requirement," *Proc. Inst. Elec. Eng., vol. 109, pt. C.*, pp. 676-684, July 1962.

[2] A. Habibi, "Hybrid coding of pictorial data," *IEEE Trans. Comm., vol. COM-22*, pp. 614-624, May 1974.

[3] S. F. Ericsson, "Fixed and adaptive predictors for hybrid predictive/transform coding," *IEEE Trans. Comm., vol. COM-33, no. 12*, pp 1291-1301, Dec. 1985.

[4] A. Habibi, "Survey of adaptive image coding techniques," *IEEE Trans. Comm., vol. COM-25*, pp. 1275-1284, Nov. 1977.

[5] J. W. Roach and J. K. Aggarwal, "Determining the movement of objects from a sequence of images," *IEEE Trans. Pattern Anal. Mach. Intell., vol. PAMI-2, no. 6*, pp. 554-562, Nov. 1980.

[6] R. Jain, "Direct computation of the focus of expansion," *IEEE Trans. Pattern Anal. Mach. Intell., vol. PAMI-5*, pp. 58-64, Jan. 1983.

[7] I. K. Sethi and R. Jain, "Finding trajectories of feature points in a monocular image sequence," *IEEE Trans. Pattern Anal. Mach. Intell., vol. PAMI-9, no. 1*, pp. 56-73, Jan. 1987.

[8] A. J. Seyler, "Visual communication and psychophysics of vision," *Proc. IRE (Australia), vol. 23*, pp. 291-304, March 1962.

[9] C. F. Hall, "Subjective evaluation of a perceptual quality metric," *SPIE, Image Quality, vol. 310*, pp. 200-204, 1981.

[10] Z. L. Budrikis, "Visual fidelity criterion and modelling," *Proc. IEEE, vol. 60, no. 7*, pp. 771-779, July 1972.

[11] J. L. Mannos and D. J. Sakrison, "The effects of a visual fidelity criterion on the encoding of images," *IEEE Trans. Inform. Theory, vol. IT-20, no. 4*, July 1974.

[12] U. Raff, D. N. Stroud, and W. Hendee, "Improvement of lesion detection in scintigraphic images by SVD techniques for resolution recovery," *IEEE Trans. Med. Imaging, vol. MI-5, no. 1*, pp. 35-44, March 1986.

[13] E. Dubois and S. Sabri, "Noise reduction in image sequences using motion-compensated temporal filtering," *IEEE Trans. Comm., vol. COM-32, no. 7*, July 1984.

[14] C. D. Bowling and R. A. Jones, "Motion compensated image coding with a combined maximum a posteriori and regression algorithm," *IEEE Trans. Comm., vol. COM-33, no. 8*, Aug. 1985.

[15] T. J. Broida and R. Chellappa, "Estimation of object motion parameters from noisy images," *IEEE Trans. Pattern Anal. and Mach. Intell., vol. PAMI-8, no. 1*, Jan. 1986.

[16] J. R. Jain and A. K. Jain, "Displacement measurement and its application in interframe image coding," *IEEE Trans. Comm., vol. COM-29, no. 12*, Dec. 1981.

[17] J. O. Limb and J. A. Murphy, "Measuring the speed of moving objects from television signals," *IEEE Trans. Comm., vol. COM-23*, pp. 474-478, Apr. 1975.

[18] C. Cafforio and F. Rocca, "Method for measuring small displacements of television images," *IEEE Trans. Inform. Theory, vol. IT-22*, pp. 573-579, Sept. 1976.

[19] H. C. Andrews and C. L. Patterson, "Singular value decomposition and their uses in digital image processing," *IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-24*, pp. 26-53, 1976.

[20] T. S. Huang, "Topics in applied physics," *Picture Processing and Digital Filtering, vol. 6*. New York: Springer-Verlag, 1975.